Web Exercise 07: Creating Project Videos and Topic Modeling

DUE Date: November 17, 2020, 5:30pm (on Blackboard).
Grade: 10 points (1.5 weeks)

Part 1: How to Create Videos using Camtasia or ZOOM (video production software).

One of your final group project requirement is to create a group project video (3 minutes) using the Camtasia or ZOOM, or any video editing software.

The Camtasia software is one of the most popular tools for creating video tutorials and presentations via screencasts or slides. (https://en.wikipedia.org/wiki/Camtasia_Studio) You can download the trial version of the Camtasia.

1. Learn How to use Camtasia with these tutorials:
https://www.techsmith.com/tutorial-camtasia.html

Please complete all ten videos (Record-Edit-Share, Video Editing Basics , Editing Audio, Produce & Share, and so on) around 50 minutes total. Then you can try to use the Camtasia by yourself in your project.

There are some other resources about Camtasia Tutorial on YouTube: For example:
https://www.youtube.com/watch?v=x1s5GWW4vlI

You can also use the ZOOM or other online tools to record your own narrative and then edit the contents.
https://www.learnupon.com/blog/record-zoom-meeting/
https://www.youtube.com/watch?v=5qnL20RveCQ
(If you know a better free Video capture and editing software, please email mtsou@sdsu.edu)

Please create a very short video (10 - 60 seconds) and publish on a YouTube Account. (Such as how you create a Carto Map or using ArcGIS online). Post the YouTube Link in the Blackboard (Under Discussion → "Video Creation Exercise (YouTube Examples)).

Note: Your Gmail account should already link to a YouTube Account. You can upload the video using your YouTube account (the same Gmail account). Make the demo video as "public" or "unlisted". (Both types can be accessed by everyone if the URL is provided).

Part 2: Topic Modeling using LDA

(acknowledgement: this part of exercise was developed by Dr. Tao Cheng and Juntao Lai from the SpaceTime Lab at UCL https://www.ucl.ac.uk/spacetimelab)

1. Introduction

In this part, you will learn how to use R to do topic analysis on Twitter data. After cleaning and formatting the raw text, you will learn how to generate topics from it using an unsupervised topic model called **Latent Dirichlet Allocation (LDA)** (Blei, 2003). LDA is a widely used method in research fields of natural language processing, text mining and social media mining. First, open the **R-studio** from your local computer. Then follow the steps below.

Text highlighted in blue is the R code, and those in green are notes.

2. Preparation

- The packages required in this exercise are:

"plyr", "stringr", " tm", "SnowballC", "lda", "LDAvis"

The "plyr" package offer functions that help to process data efficiently. The "stringr" and "tm" package are widely used in text mining, which provides a series of functions of processing and formatting text. The "SnowballC" is used to stem documents, which will be explained in section 3. The "lda" package enables us to fit the data with LDA model. The "LDAvis" package offers an interactive visualisation of outputs of the topic model. It will be explained in more detail in section 6.

```
# Install the R packages
install.packages(c("plyr","stringr","tm","SnowballC","lda","LDAvis"))

library("plyr")
library("stringr")
library("tm")
library("SnowballC")
library("lda")
library("LDAvis")
```

Twitter data

A sample of Geotagged Twitter data in London in one week time period (2016-06-27 to 2016-07-03) is provided for this exercise. Please download it from the Google Shared Drive (TwitterSample.csv)

The next step is to import the CSV file into R and have a look at it. It includes 5 variables, "USERID", "LAT", "LON", "STATUE_DATE" and "TWEET_TEXT". These variables allow us to analyze Twitter data spatially and temporally.

Moreover, we can generate **semantic topics from the content**. During this exercise, you will learn how to generate topics from the raw Tweets.

```
# set the work directory to where the data been stored in your local disk

setwd("D:/yourlocalworkdirectory//")

#load Twitter data sample

Twitter<-read.csv("TwitterSample.csv")
```

3. Text Cleaning and formatting

The quality of input data has great impact on the results of the topic model. Unlike other word documents such as papers and blogs, Twitter data contains tons of non-standard words, expressions and symbols. Before taking any further step, **the text of Tweets should be cleaned and formatted.** This requires knowledge of Natural Language Processing. This tutorial presents the basic approaches for you to start with. You may get a warning message while you're running the script, but you can ignore it since it is not an error message.

At first, we create a corpus with the Tweet texts. A corpus is a large and structured set of texts. The text cleaning processes will be applied on this data structure.

```
Tweets_corpus <- Corpus(VectorSource(Twitter$TWEET_TEXT))
```

With the "**tolower**" function, capitalized letters are converted into lower case.

```
Tweets_corpus <- tm_map(Tweets_corpus, tolower)
```

You can check the changes of the text using `> Tweets_corpus[[1]]` after each step.
The following code is used to make the text clean and simple:

```
# remove punctuation

Tweets_corpus <- tm_map(Tweets_corpus, removePunctuation)

# remove numbers

Tweets_corpus <- tm_map(Tweets_corpus, removeNumbers)
```

```
# remove URLs

Tweets_corpus <- tm_map(Tweets_corpus, function(x) gsub("http[[:alnum:]]*","", x))

# remove NonASCII characters

Tweets_corpus <- tm_map(Tweets_corpus, function(x) iconv(x, "latin1", "ASCII", sub=""))
```

However, the choice of these steps may vary according to the purpose of your application. For example, some numbers could be meaningful in some situations such as 911 for emergency.

In order to get more meaningful key words for the following analysis, the next step is **to remove the stopwords.** Stopwords are defined as the words which are filtered before the processing of text data. Stopwords are usually the most common and short function words, such as "is", "in" and "the". Here we use the English stop words from the SMART information retrieval system (System for the Mechanical Analysis and Retrieval of Text) which is available in the package "tm". (*Note: The SMART (System for the Mechanical Analysis and Retrieval of Text) Information Retrieval System is an information retrieval system developed at Cornell University in the 1960s. Many important concepts in information retrieval were developed as part of research on the SMART system, including the vector space model, relevance feedback, and Rocchio classification.*)

```
# remove stopwords

Tweets_corpus <- tm_map(Tweets_corpus, removeWords,stopwords("SMART"))
```

Similarly, the user can define a word list to be removed apart from the stopwords.

```
# remove specific words

Tweets_corpus <- tm_map(Tweets_corpus, removeWords,c("london", "im","ive", "dont", "didnt"))
```

After that, the white spaces need to be formatted.

```
Tweets_corpus <- tm_map(Tweets_corpus, stripWhitespace)
```

Then, the process of "stemming" is applied to reduce inflected words to their stem form, by removing suffixes of the words, such as –ed, -ing and -ly. This prevents words with same meaning from been treated as different key words.

```
Tweets_corpus <- tm_map(Tweets_corpus, PlainTextDocument)
Tweets_corpus <- tm_map(Tweets_corpus, stemDocument)
```

Now that the text has been cleaned, we can convert the data structure from corpus back to a character list and remove the extra whitespaces.

```
# unlist the text corpus

Tweet_Clean<-as.data.frame(unlist(sapply(Tweets_corpus[[1]]$content,'[')), stringsAsFactors=F)

# remove extra whitespace in text

Tweet_Clean <- lapply(Tweet_Clean[,1], function(x) gsub("^ ", "", x)) #multiple spaces
Tweet_Clean <- lapply(Tweet_Clean, function(x) gsub("^[[:space:]]+", "", x)) #space at the begining
Tweet_Clean <- lapply(Tweet_Clean, function(x) gsub("[[:space:]]+$", "", x)) #space at the end
```

The cleaned text can be bound to the original Twitter data, so that you can compare them to see the changes.

```
# bind clean text with Twitter data

Twitter$Tweet_Clean<-Tweet_Clean

# check the first 10 Tweets

Twitter[1:10,]
```

4. Preparation for topic modelling

In the previous steps, we have our text cleaned and formatted. Before fitting a topic model, the text needs to be tokenized. **Tokenization is the task of chopping the sentences into words** (called tokens).

```
# tokenize on space and output as a list:

doc.list <- strsplit(unlist(Tweet_Clean), "[[:space:]]+")
```

Now, we can compute the frequency of each term and remove those low frequencies terms. As many words in Tweets are informal, applying this approach can help to remove them so subsequent processes are easier.

```
# compute the table of terms:
term.table <- table(unlist(doc.list))
term.table <- sort(term.table, decreasing = TRUE)

# remove terms that are stop words or occur fewer than 3 times:
```

```
term.table <- term.table[term.table>3]
```

Then create a vocabulary with remaining terms.

```
vocab <- names(term.table)
```

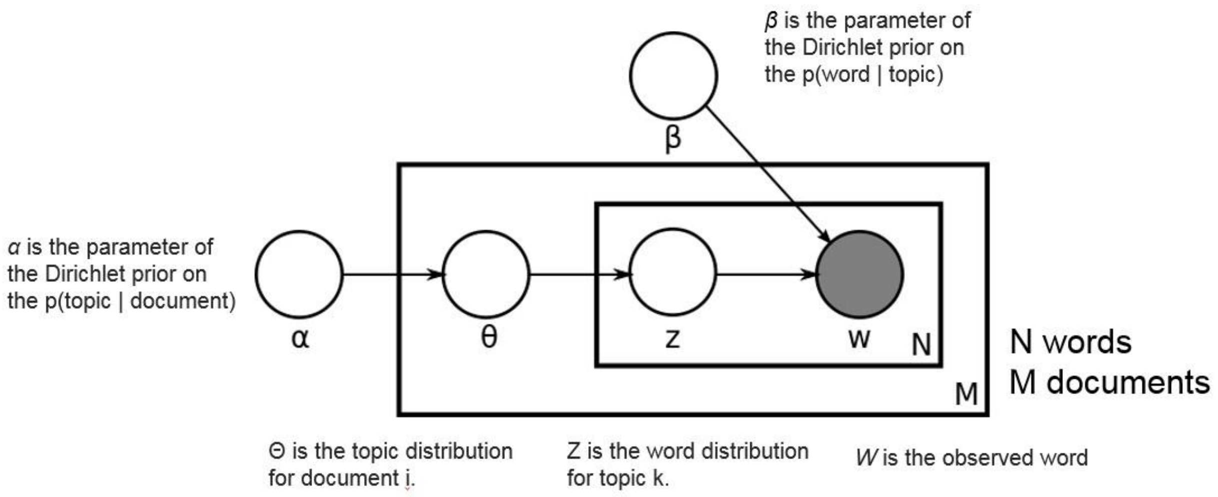
Run the following codes to put the documents into the format required by **the lda package**. The object "documents" is a list where each element represents one document (Tweet).

```
# put the documents into the format required by the lda package:

get.terms <- function(x) {

   index <- match(x, vocab)

   index <- index[!is.na(index)]

   rbind(as.integer(index - 1), as.integer(rep(1, length(index))))

}

documents <- lapply(doc.list, get.terms)
```

We can also compute some statistics of our input dataset. These statistics can be used for visualization afterwards.

```
# Compute some statistics related to the data set:
D <- length(documents)  # number of documents

W <- length(vocab)  # number of terms in the vocab

doc.length <- sapply(documents, function(x) sum(x[2, ]))  # number of tokens per document
N <- sum(doc.length)  # total number of tokens in the data

term.frequency <- as.integer(term.table) # frequencies of terms in the corpus
```

5. Fit LDA topic model

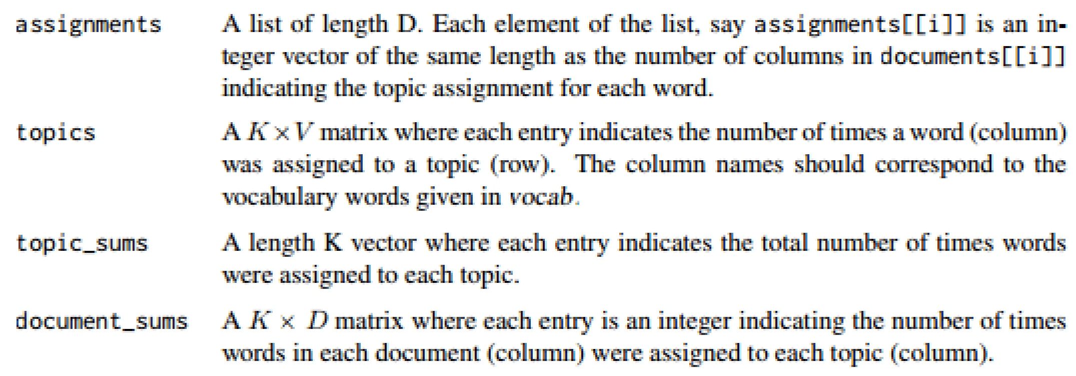

β is the parameter of the Dirichlet prior on the p(word | topic)
β
α is the parameter of the Dirichlet prior on the p(topic | document)
α
θ
z
w
N
M
N words
M documents
Θ is the topic distribution for document i.
Z is the word distribution for topic k.
W is the observed word

In this practical, we use the "lda" R package produced by Jonathan Chang. We set up a topic model with **K=20 topics,** the priors for the topic-term distributions (β = 0.1) and document-topic distributions (α = 0.1). The choice of these inputs depends on the application. Basically, **bigger α means more similarity between the topics in different documents** and **bigger β means more similarity between the words in different topics.** We set the collapsed Gibbs sampler to run for G=1,000 iterations to ensure convergence. It will take a while to fit the model, you can use "Sys.time()" to check the time consumed by this process.

```
### fit LDA model

# parameters

K <- 20

G <- 1000

alpha <- 0.1

eta <- 0.1


t1 <- print(Sys.time())

lda_fit <- lda.collapsed.gibbs.sampler (documents = documents, K = K, vocab = vocab,  num.iterations =
G, alpha = alpha, eta = eta)

t2 <- print(Sys.time())

t2-t1
```

6. LDA outputs

As the output of the fitted lda model, the object "lda_fit" is a list storing several components, we focus on the following 4 components.



`assignments`	A list of length D. Each element of the list, say `assignments[[i]]` is an integer vector of the same length as the number of columns in `documents[[i]]` indicating the topic assignment for each word.
`topics`	A $K \times V$ matrix where each entry indicates the number of times a word (column) was assigned to a topic (row). The column names should correspond to the vocabulary words given in *vocab*.
`topic_sums`	A length K vector where each entry indicates the total number of times words were assigned to each topic.
`document_sums`	A $K \times D$ matrix where each entry is an integer indicating the number of times words in each document (column) were assigned to each topic (column).

You can first quickly examine the topics by the function "top.topic.words", which will return a matrix of the top words in each topic. The number 20 here means top 20 words of each topic will be shown.

```
top_words<-top.topic.words(lda_fit$topics,20,by.score=TRUE)
```

Have a look at the table "top_words", each column presents top 20 words of that topic. Does the result make sense to you? Can you interpret the topics according to this?

With the help of the R package "LDAvis" (produced by Carson Sievert and Kenny Shirley), we can also create an interactive visualisation of the topics. To visualize the result using LDAvis, we'll need estimates of the document-topic distributions, which we denote by the D×K matrix θ, and the set of topic-term distributions, which we denote by the K×W matrix φ.

```
theta <- t(apply(lda_fit$document_sums + alpha, 2, function(x) x/sum(x)))

phi <- t(apply(t(lda_fit$topics) + eta, 2, function(x) x/sum(x)))
```

We've already computed the number of tokens per document and the frequency of the terms across the entire corpus. We save these, along with φ, θ, and vocab, in a list as the data object "Tweet_Topics".

```
 Tweet_Topics <- list(phi = phi, theta = theta, doc.length = doc.length,  vocab = vocab,
                      term.frequency = term.frequency)
```

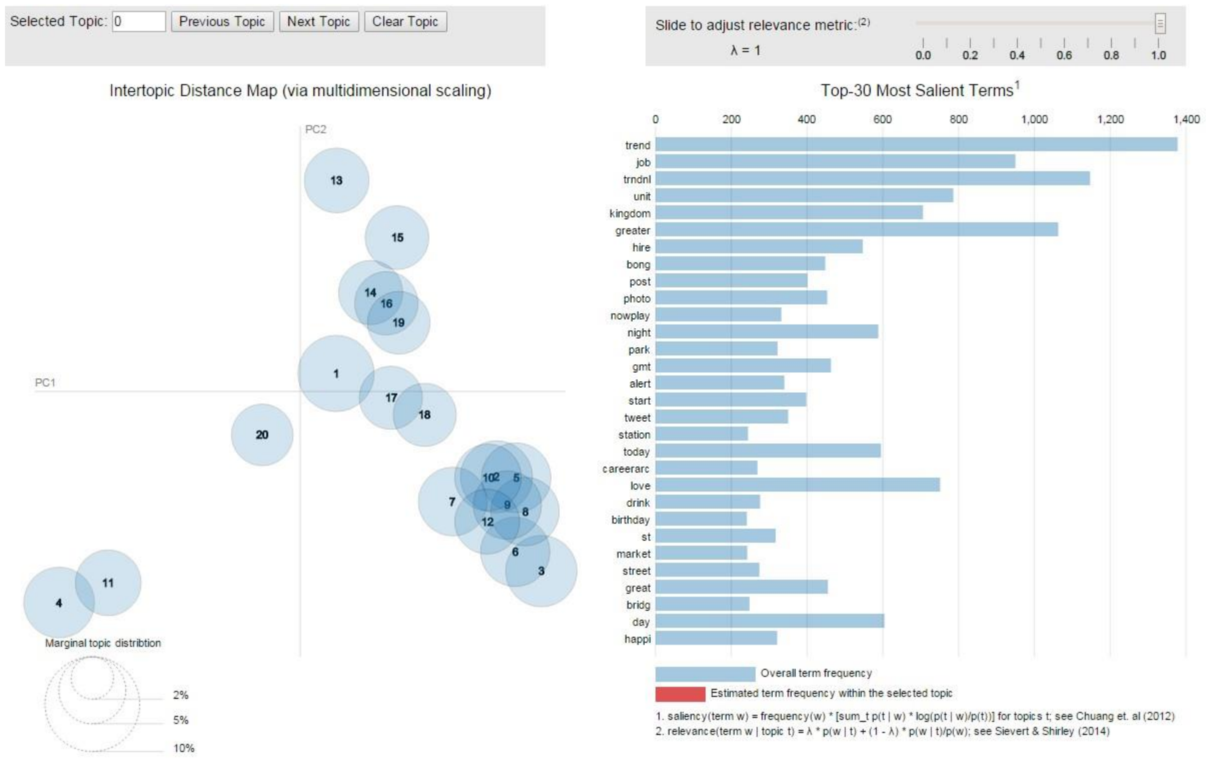
Then we used the function "createJSON()" to have a JSON object which to be used for visualization. The function will compute topic frequencies, inter-topic distances and project topics onto a 2D plane to represent their similarity to each other.

```
Tweet_Topics_json <- with(Tweet_Topics,createJSON(phi, theta, doc.length, vocab, term.frequency))
```

After that, you can use the function “serVis” to display the results as an interactive figure.

```
serVis(Tweet_Topics_json)
```

There is a tuning parameter ($0 \leqslant \lambda \leqslant 1$) that controls how the terms are ranked for each topic, where terms are listed in decreasing of relevance. Values of λ near 1 give high relevance rankings to frequent terms within a given topic, whereas values of λ near zero give high relevance rankings to exclusive terms within a topic.



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

7. Assign topic for each tweet

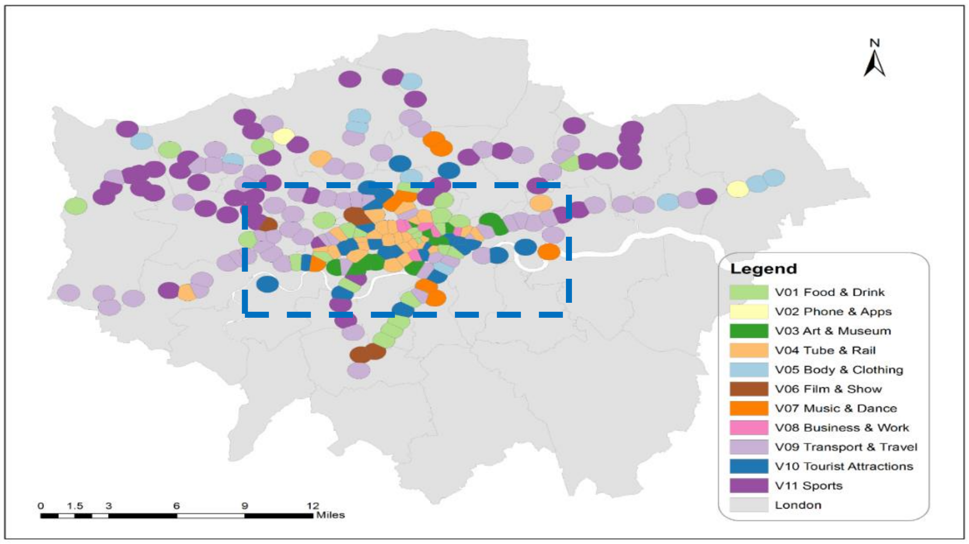
The topic model can **return back the probabilities of each word belong to different topics.** Within a Tweet, different words carry different topics. However, to make it simple in this tutorial, the topic that appears most frequently within that Tweet will be the topic that the Tweet is assigned to.

```
doc_topic <- apply(lda_fit$document_sums, 2, function(x) which(x == max(x))[1])

Twitter$topic<-doc_topic
```

Now, each Tweet has its dominant topic. If you would like to look at the spatial distribution of each topic, you can export the object "Twitter" and display it in ArcMap.

```
TwitterDf <- data.frame(lapply(Twitter, as.character), stringsAsFactors=FALSE)

write.csv(TwitterDf,"TwitterWithTopic.csv")
```

Please save this **TwitterWithTopic.csv** into your local disk. We will visualize this file using ArcGIS Online Table next.



8. Summary

In this workshop, we have learnt how to do basic text cleaning in R and how to generate topics using LDA topic model. Based on the results, you can explore the spatial-temporal pattern of Tweets in different topics. If you would like to learn more about topic modelling, there are many good references online. For examples:

Introduction to LDA:

http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/

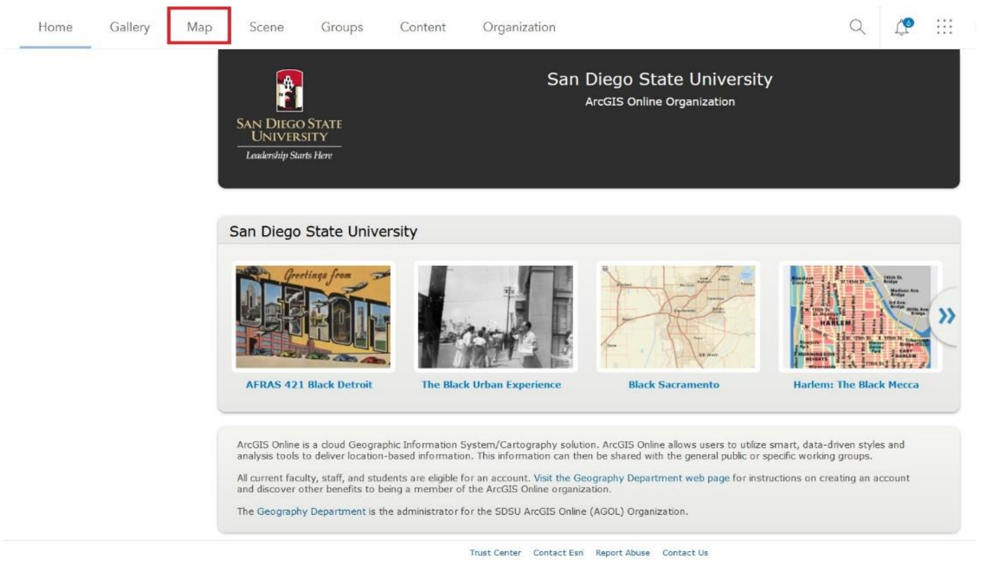
Theory of LDA:
Blei, David M., Andrew Y. Ng, and Michael I. Jordan.(2003) "Latent dirichlet allocation." the Journal of Machine Learning Research 3: 993-1022.

Visualizing the output of LDA:
http://cpsievert.github.io/LDAvis/reviews/reviews.html

Part 3: Visualizing CSV files using ArcGIS Online

ArcGIS Online is a web application allowing sharing and search of geographic information, as well as content published by Esri, ArcGIS users, and other authoritative data providers. It allows users to create and join groups, and control access to items shared publicly or within groups. (cited from Wikipedia).

- 1. Go to https://sdsugeo.maps.arcgis.com
  2. Click the "Sign In" dropdown in the upper right
  3. Select the blue "San Diego State University" button - this take you to the SDSU ID login window
  4. Use your SDSU ID username and password.
  5. When you login, go to the "Map" menu on the top.

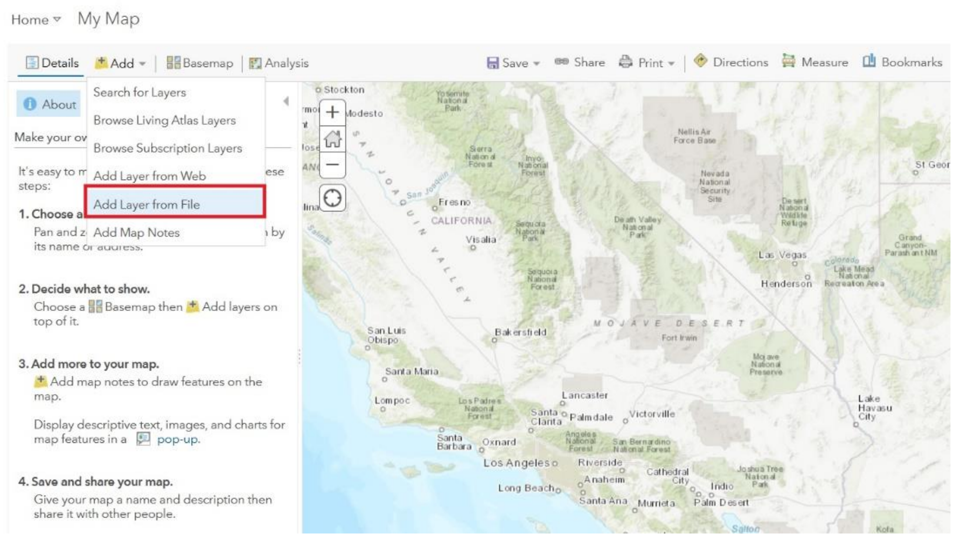


ArcGIS Online is a cloud Geographic Information System/Cartography solution. ArcGIS Online allows users to utilize smart, data-driven styles and analysis tools to deliver location-based information. This information can then be shared with the general public or specific working groups.

All current faculty, staff, and students are eligible for an account. Visit the Geography Department web page for instructions on creating an account and discover other benefits to being a member of the ArcGIS Online organization.

The Geography Department is the administrator for the SDSU ArcGIS Online (AGOL) Organization.

- **Import the TwitterSample.csv file from your local folder directly.**



Make your ov

It's easy to m ... ese steps:

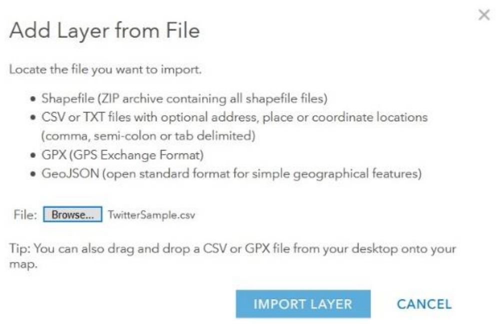
1. Choose a ... Pan and z ... by its name or address.

2. Decide what to show. Choose a Basemap then Add layers on top of it.

3. Add more to your map. Add map notes to draw features on the map.

Display descriptive text, images, and charts for map features in a pop-up.

4. Save and share your map. Give your map a name and description then share it with other people.



Add Layer from File

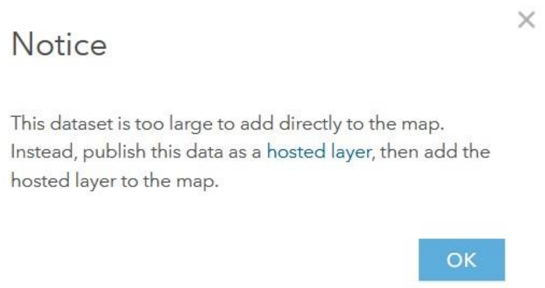
Locate the file you want to import.

- Shapefile (ZIP archive containing all shapefile files)
- CSV or TXT files with optional address, place or coordinate locations (comma, semi-colon or tab delimited)
- GPX (GPS Exchange Format)
- GeoJSON (open standard format for simple geographical features)

File: Browse... TwitterSample.csv

Tip: You can also drag and drop a CSV or GPX file from your desktop onto your map.

IMPORT LAYER CANCEL

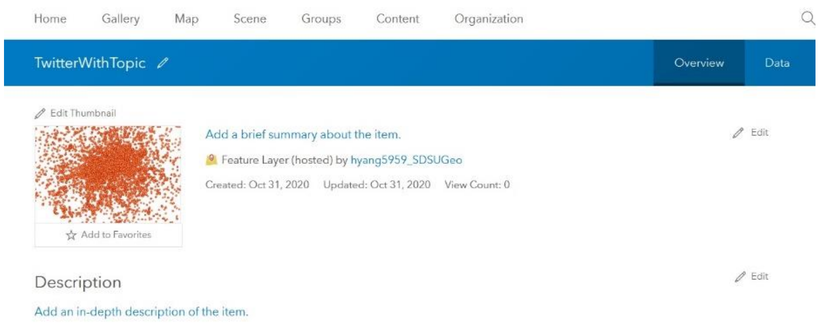


Notice

This dataset is too large to add directly to the map. Instead, publish this data as a hosted layer, then add the hosted layer to the map.

OK

For next step you will be requested to publish the data as a hosted layer in advance due to the size of the dataset. Click on the link "hosted layer" on the pop-up window and follow the steps of "Publish a CSV file" (https://doc.arcgis.com/en/arcgis-online/manage-data/publish-features.htm).



TwitterWithTopic

Add a brief summary about the item.

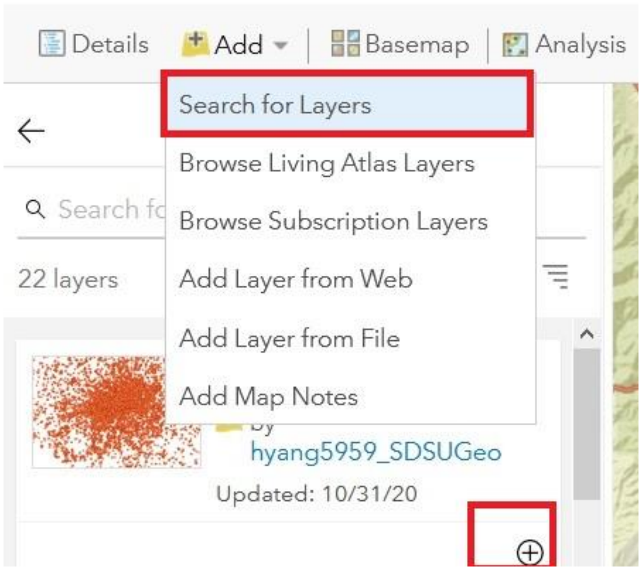
Feature Layer (hosted) by hyang5959_SDSUGeo

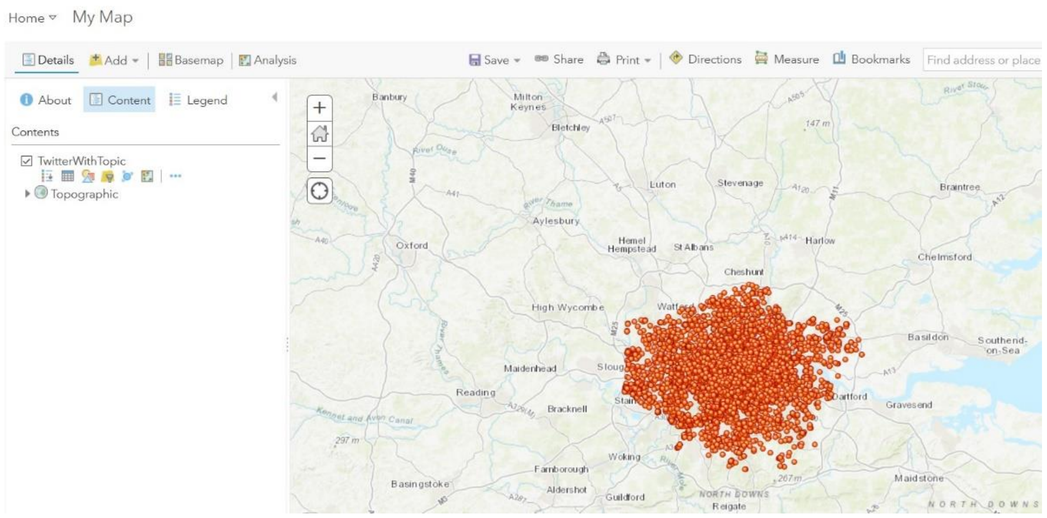
Created: Oct 31, 2020 Updated: Oct 31, 2020 View Count: 0

Description

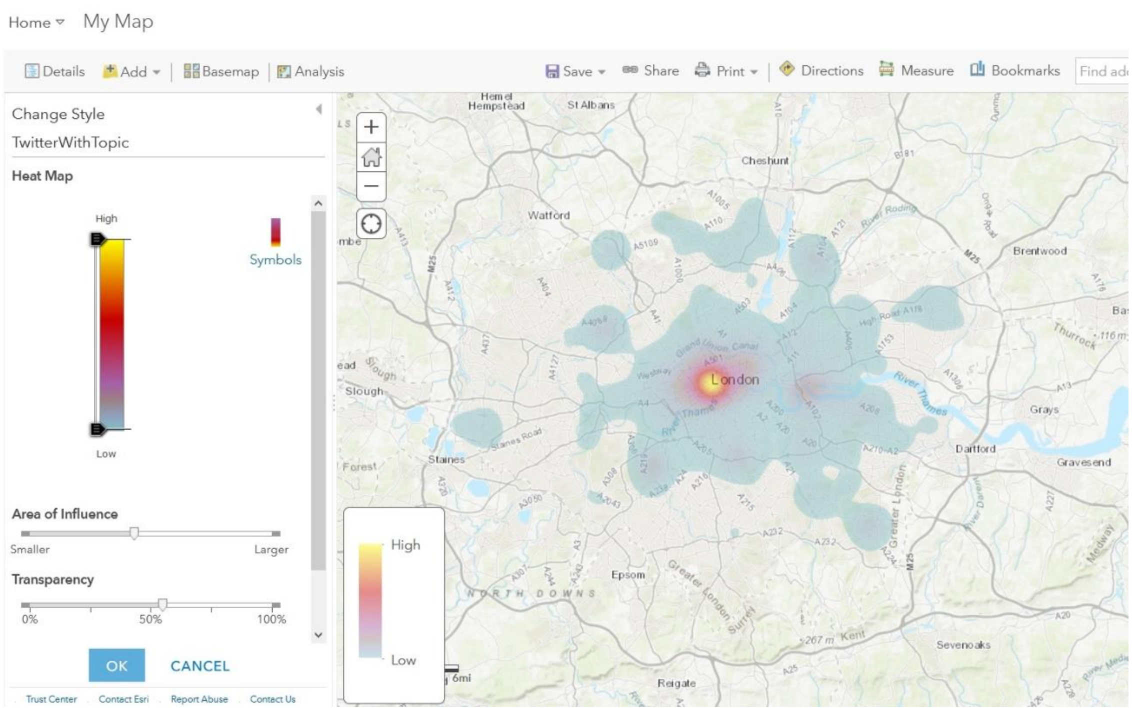
Add an in-depth description of the item.

Then go back to "Map" and add the layer that you created.





- **You can also try the "Heat map" function to provide different types of visualization.**



Now, try to import the "TwitterWithTopic.csv" file into the ArcGIS online and use different color to display different topics (Hint: Content → Change style →Choose "topic" as the attribute to show and select an appropriate drawing style). Create a Screenshot of the result for this exercise.

After finishing this Web Course, Please use your own words to answer the following questions (next page): (DO NOT COPY any web resources or Wikipedia texts. We will check your answers with Blackboard tools to verify that your responses are uniquely yours.) By submitting your answers (paper) to Blackboard, you agree: (1) that you are submitting your paper to be used and stored as part of the SafeAssign™ services in accordance with the Blackboard Privacy Policy; (2) that your institution may use your paper in accordance with your institution's policies; and (3) that your use of SafeAssign will be without recourse against Blackboard Inc. and its affiliates.

SafeAssign accepts files in .doc, .docx, .docm, .ppt, .pptx, .odt, .txt, .rtf, .pdf, and .html file formats only. Files of any other format will not be checked through SafeAssign.

LAB-7 Questions:

1. **What is the YouTube URL of your short demo video? Which software do you use to create this video? What are the advantages and disadvantages of the video software you used?**

2. **What are the key procedures for cleaning texts in Twitter messages?**
3. **What is LDA? How to interpret topic-term distributions parameter (β) and document-topic distributions parameter (α)?**
4. **Include an ArcGIS Online Screenshot of the London Map with the LDA and the results of “serVis” display. (use different colors or markers to display different topics). Select one color (topic) to explain the possible represented topic.**

Please submit your LAB-7 Answers (in a MS Word or a PDF file format only) to the Blackboard System BEFORE the DUE DATE/TIME.