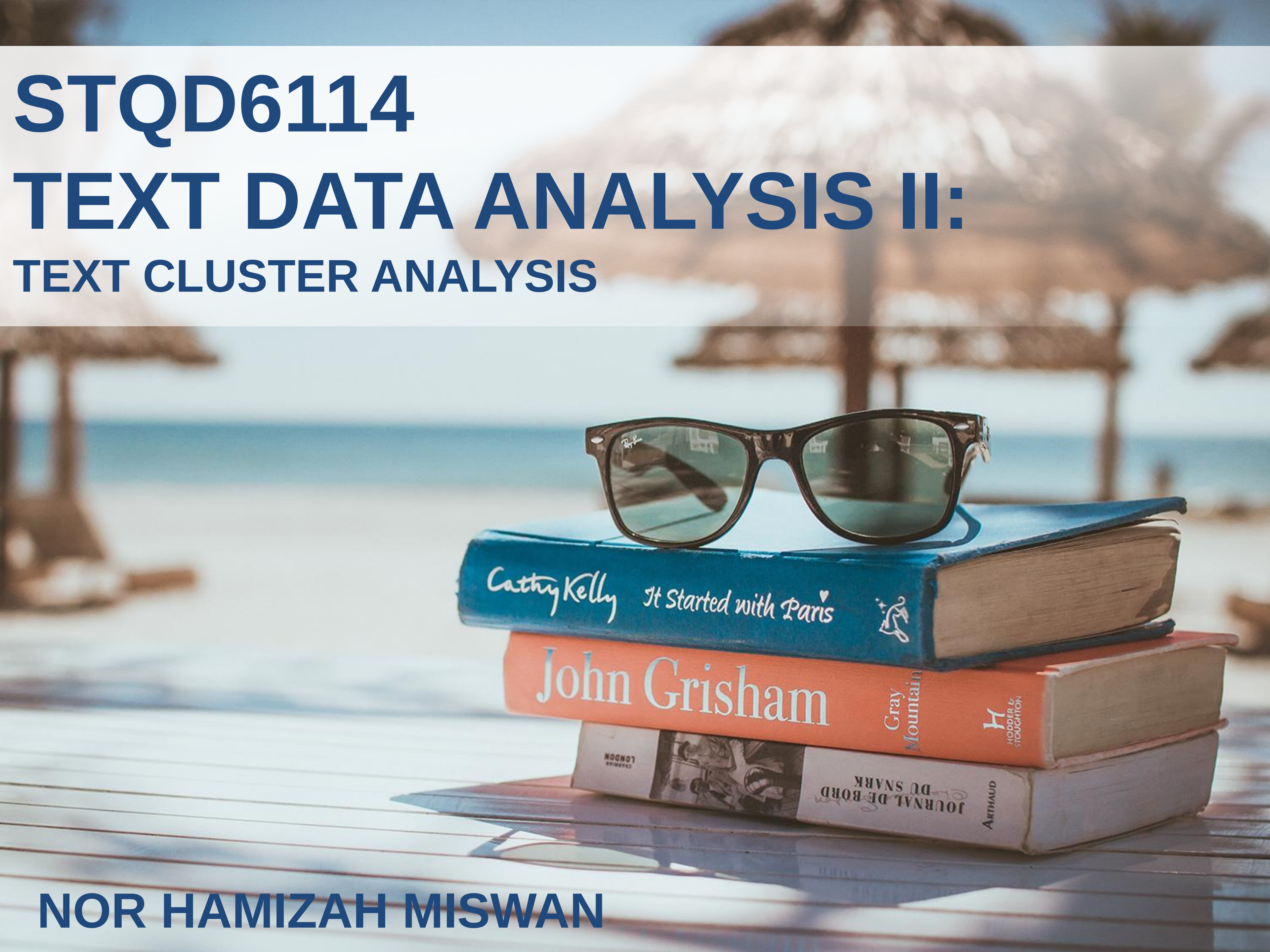


STQD6114 TEXT DATA ANALYSIS II: TEXT CLUSTER ANALYSIS

NOR HAMIZAH MISWAN

Introduction

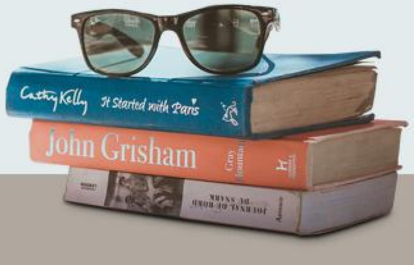
- ❖Cluster: to group, to categorize
- ❖Element / subjects that are similar should be clustered together
- ❖Elements are evaluated on the similarity or differences index
- ❖The use of distance concept

More on cluster analysis

- ❖ Task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).
- ❖ Can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them.
- ❖ Popular notions of clusters include groups with small distances [Note: we'll use distance-based methods] among the cluster members, dense areas of the data space, intervals or particular statistical distributions [i.e. distribution s of words within documents and the entire collection].

Why?

- Typically when there is a very large collection of documents, auto clustering is needed
- Hence, there is a need of algorithm in text clustering analysis



In general, the algorithm first performs a series of transformations on the free flow text data (elaborated in subsequent sections) and then performs a clustering algorithm on the vectorized form of the transformed data. Subsequently, the algorithm creates cluster-wise tags, also known as cluster-centers, that are representative of the data contained in these clusters.

The solution boasts of end-to-end automation and is generic enough to operate on any dataset.

The text clustering algorithm works in five stages enumerated below:-

- ❖ Transformations on raw stream of free flow text
- ❖ Creation of Term Document Matrix
- ❖ TF-IDF (Term Frequency – Inverse Document Frequency) Normalization
- ❖ Distance Computation and Clustering Algorithm
- ❖ Auto-Tagging based on Cluster Centers

Stage 1

Removing punctuations

Transforming to lower case

Grammatically tagging sentences and removing pre-identified stop phrases (Chunking)

Removing numbers from the document

Stripping any excess white spaces

Stage 2

Removing generic words of the English language viz. determiners, articles, conjunctions and other parts of speech.

Stage 3

Document Stemming which reduces each word to its root using Porter’s stemming algorithm.

Stage 4: TF-IDF (Term Frequency – Inverse Document Frequency) Normalization

This is an optional step and can be performed in case there is high variability in the document corpus and the number of documents in the corpus is extremely large (of the order of several million).

This normalization increases the importance of terms that appear multiple times in the same document while decreasing the importance of terms that appear in many documents (which would mostly be generic terms).

intended to reflect how important a word is to a document in a collection or corpus.

Stage 4: Compute distance & clustering algorithms

Post the TF-IDF transformation, the document vectors are put through a clustering algorithm which computes the distances amongst these documents and clusters nearby documents together.

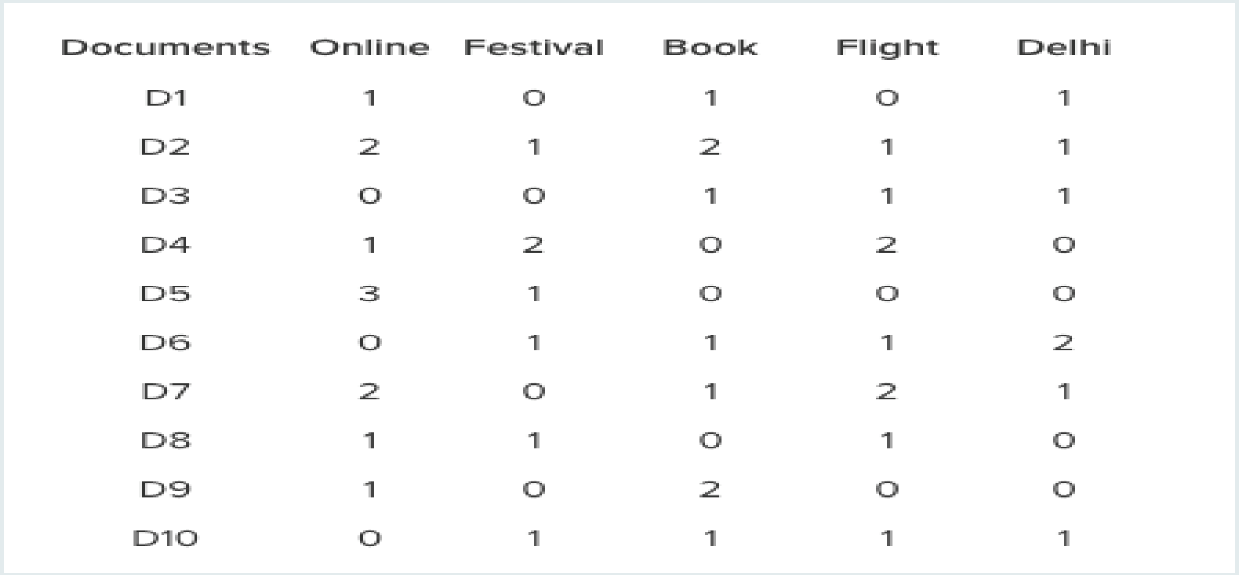
Stage 5: Auto-Tagging based on Cluster Centers

The algorithm then generates cluster tags, known as cluster centers which represent the documents contained in these clusters. The clustering and auto-generated tags are best depicted in the illustration below (Principal components 1 and 2 are plotted along the x and y axes respectively).

k-means algorithm

- The basic idea of K Means clustering is to form K seeds first, and then group observations in K clusters on the basis of distance with each of *K* seeds.
- The observation will be included in the nth seed/cluster if the distance between the observation and the nth seed is minimum when compared to other seeds.
- Next, we look on a brief example on performing a *K* Means Clustering Analysis.

Example



Documents	Online	Festival	Book	Flight	Delhi
D1	1	0	1	0	1
D2	2	1	2	1	1
D3	0	0	1	1	1
D4	1	2	0	2	0
D5	3	1	0	0	0
D6	0	1	1	1	2
D7	2	0	1	2	1
D8	1	1	0	1	0
D9	1	0	2	0	0
D10	0	1	1	1	1

- ❖ Let's assume that we want to create K=3 clusters. First, three seeds should be chosen. Suppose, D2, D5 & D7 are chosen as initial three seeds.
- ❖ The next step is to calculate the Euclidean distance of other documents from D2, D5 & D7.
- ❖ Assuming: U=Online, V= Festival, X=Book, Y=Flight, Z=Delhi. Then the Euclidean distance between D1 & D2 would be:

((U1-U2)^2 + (W1-W2)^2+(X1-X2)^2+ (Y1-Y2)^2+

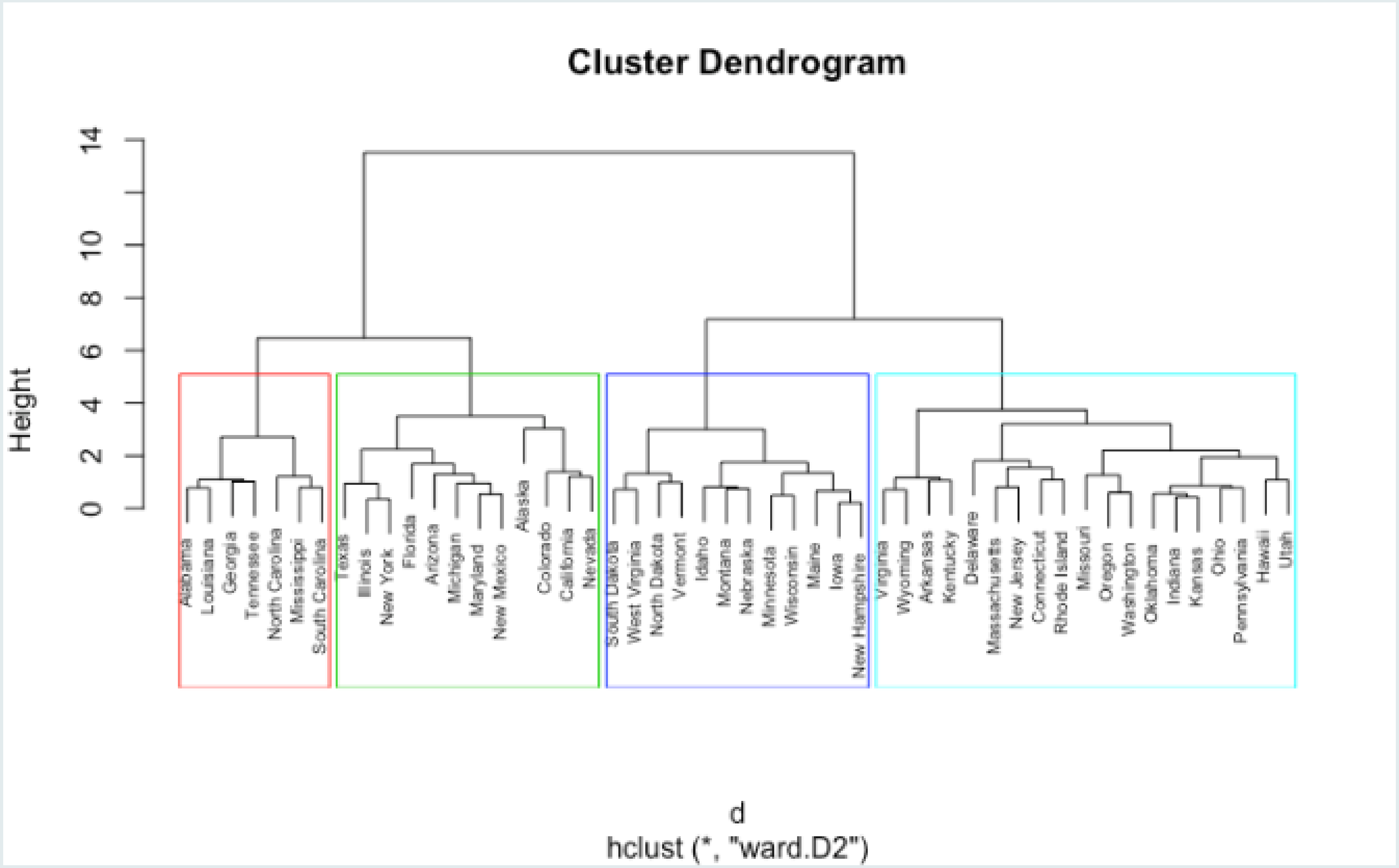
(Z1-Z2)^2)^0.5

Documents	D2	D5	D7	Min. Distance	Movement
D1	2.0	2.6	2.2	2.0	D2
D2	0.0	2.6	1.7	0.0	
D3	2.4	3.6	2.2	2.2	D7
D4	2.8	3.0	2.6	2.6	D7
D5	2.6	0.0	2.8	0.0	
D6	2.4	3.9	2.6	2.4	D2
D7	1.7	2.8	0.0	0.0	
D8	2.6	2.0	2.8	2.0	D5
D9	2.0	3.0	2.6	2.0	D2
D10	2.2	3.5	2.4	2.2	D2

Clusters	# of Observations
D2	5
D5	2
D7	3

Hierarchical clustering

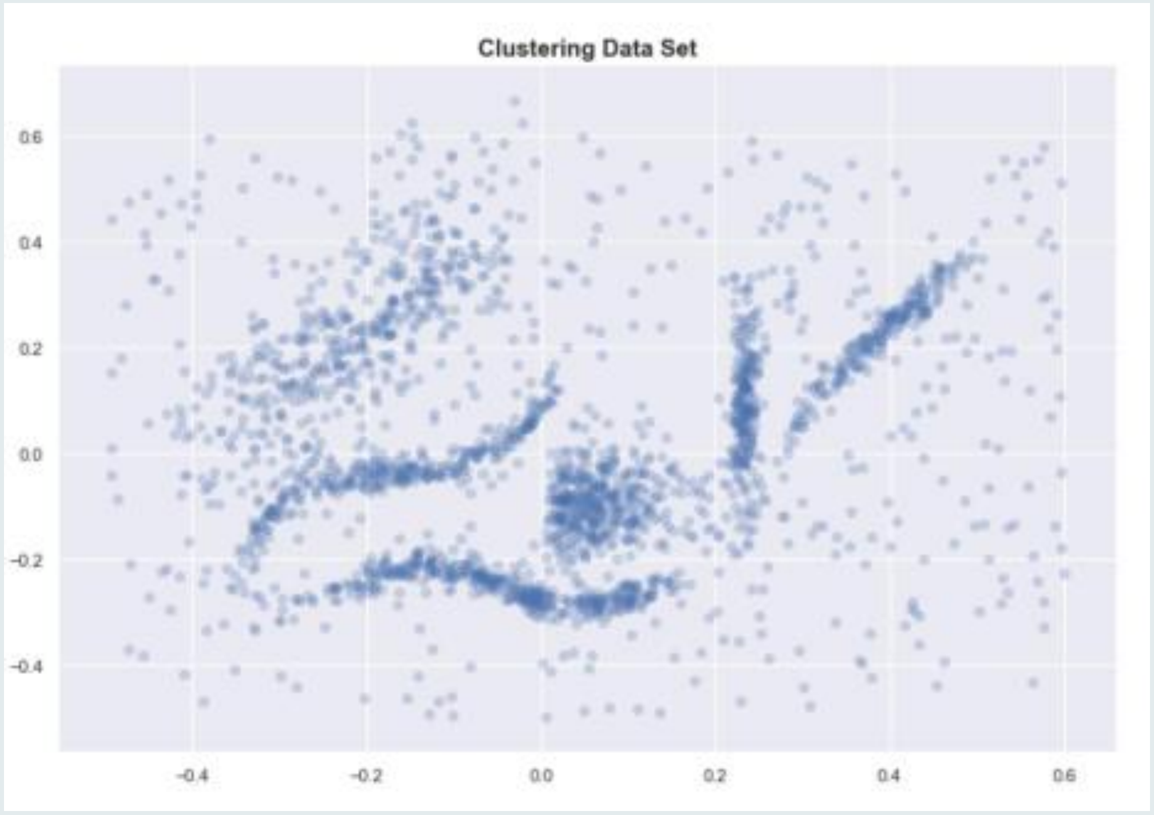
- Bottoms up
- Complete, average, single, Ward
- Dendrogram
- Start with all documents in their individual cluster, and regroup the documents based on the algorithm

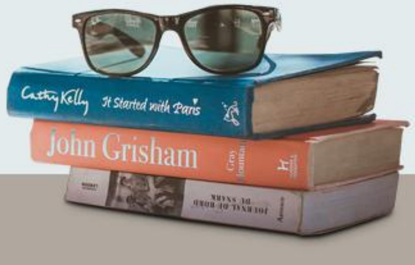


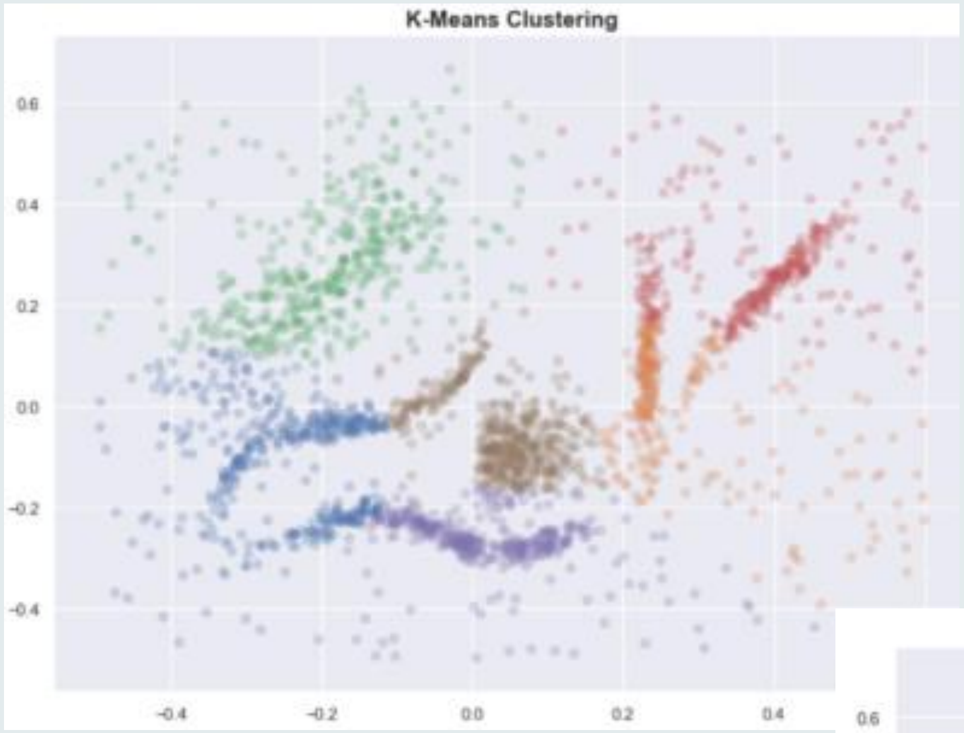
Cluster Dendrogram
Height
0
2
4
6
8
10
14
Alabama
Louisiana
Georgia
Tennessee
North Carolina
Mississippi
South Carolina
Texas
Illinois
New York
Florida
Arizona
Michigan
Maryland
New Mexico
Alaska
Colorado
California
Nevada
South Dakota
West Virginia
North Dakota
Vermont
Idaho
Montana
Nebraska
Minnesota
Wisconsin
Maine
Iowa
New Hampshire
Virginia
Wyoming
Arkansas
Kentucky
Delaware
Massachusetts
New Jersey
Connecticut
Rhode Island
Missouri
Oregon
Washington
Oklahoma
Indiana
Kansas
Ohio
Pennsylvania
Hawaii
Utah
d
hclust (*, "ward.D2")

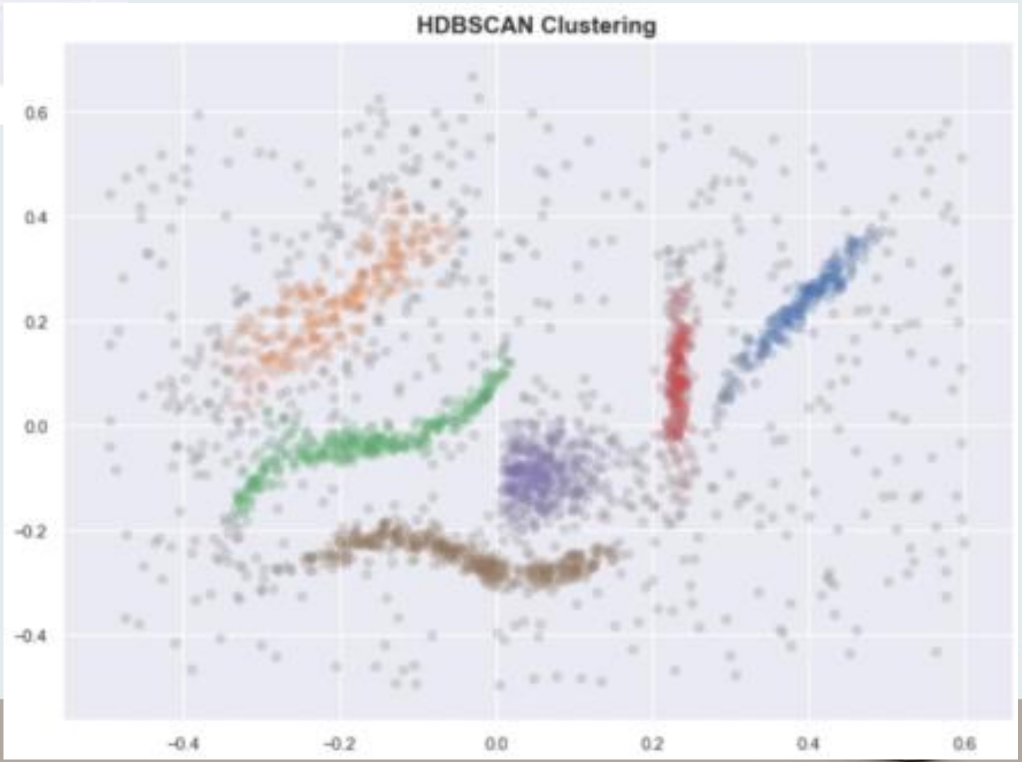
Density based HDBScan

- ❖ Hierarchical Density-Based Spatial Clustering of Applications with Noise



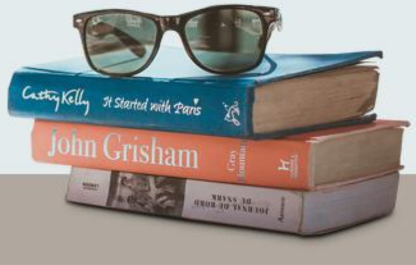



K-Means Clustering
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.4
-0.2
0.0
0.2
0.4

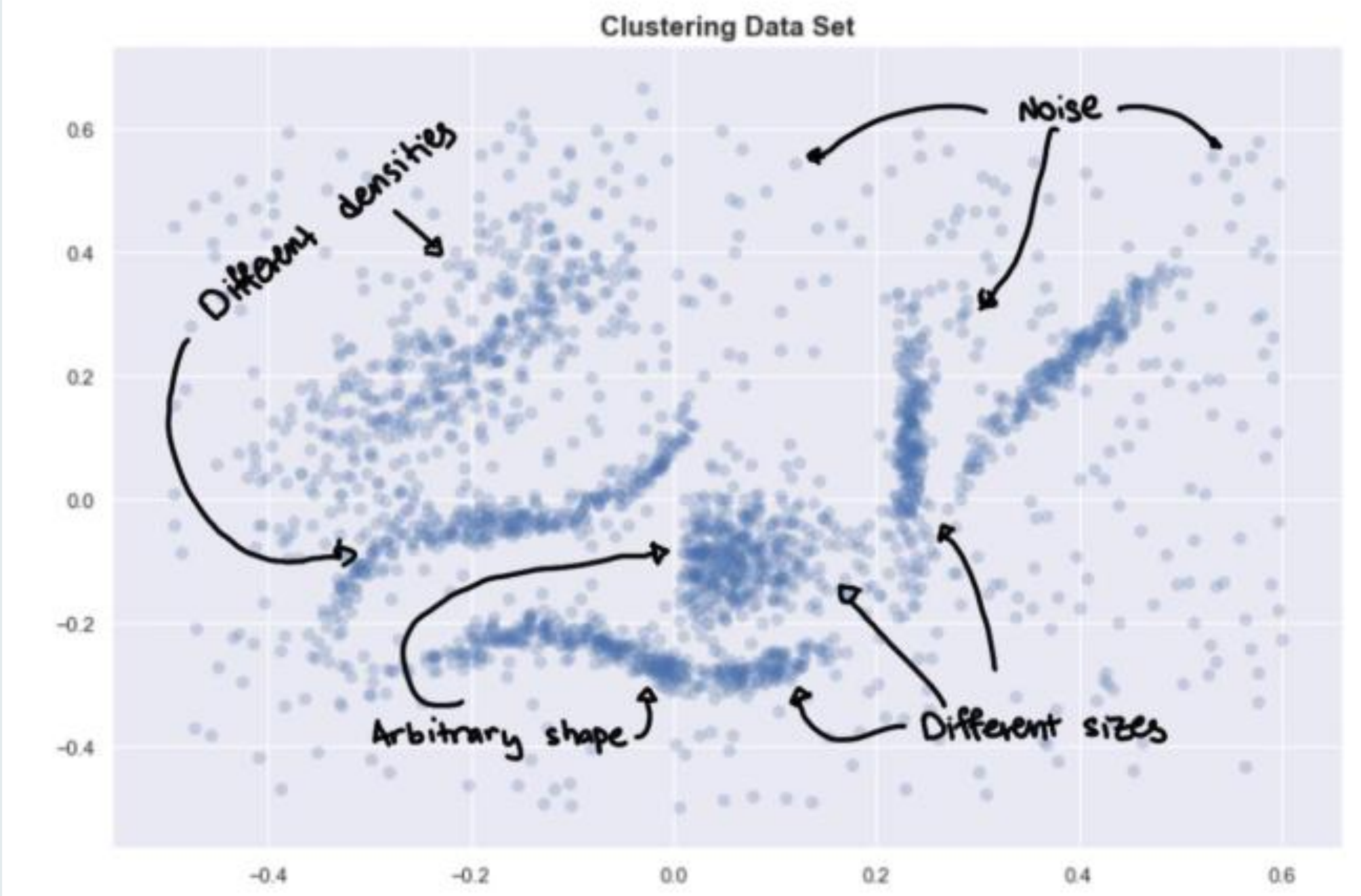

HDBSCAN Clustering
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.4
-0.2
0.0
0.2
0.4
0.6

We go back to our original data set and by simply describing it, it becomes obvious why *K*-means has a hard time. The data set has:

- ❖ Clusters with arbitrary shapes
- ❖ Clusters of different sizes
- ❖ Clusters with different densities
- ❖ Some noise and maybe some outliers



- The core ideas of HDBSCAN and how it excels even when the data has:
  - Arbitrarily shaped clusters
  - Clusters with different sizes and densities
  - Noise
- HDBSCAN uses a density-based approach which makes few implicit assumptions about the clusters.
- It is a non-parametric method that looks for a cluster hierarchy shaped by the multivariate modes of the underlying distribution.
- Rather than looking for clusters with a particular shape, it looks for regions of the data that are denser than the surrounding space.
- The mental image you can use is trying to separate the islands from the sea or mountains from its valleys.



Clustering Data Set
Different densities
Noise
Arbitrary shape
Different sizes
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.4
-0.2
0.0
0.2
0.4
0.6

Cathy Kelly It Started with Paris
John Grisham

Text clustering vs Topic modelling

	TC	TM
Aim	To group the documents into coherent categories	To find underlying themes in a group documents
Approach	Use similarity measures	Use hierarchical probabilistic distributions
Output	a list of clusters with every document show ing up in one of the clusters	a list of topics with associated clusters of words.

Appendix

```
mytext<-DirSource("TextFile")
docs<-VCorpus(mytext)

docs <- tm_map(docs,content_transformer(tolower))
toSpace <- content_transformer(function(x, pattern) { return (gsub(pattern, " ", x))})
docs <- tm_map(docs, toSpace, "-")
docs <- tm_map(docs, toSpace, ":")
docs <- tm_map(docs, toSpace, "'")
docs <- tm_map(docs, toSpace, ".")
docs <- tm_map(docs, toSpace, ".     ")
docs <- tm_map(docs, toSpace, " -")
docs <- tm_map(docs, removePunctuation) #remove punctuation
docs <- tm_map(docs, removeNumbers) #Strip digits
docs <- tm_map(docs, removeWords, stopwords("english")) #remove stopwords
docs <- tm_map(docs, stripWhitespace) #remove whitespace
```

```
tdm <- DocumentTermMatrix(docs) #Create document term matrix
tdm

#Present text data numerically, weighted TF-IDF
tdm.tfidf <- weightTfIdf(tdm)
tdm.tfidf <- removeSparseTerms(tdm.tfidf, 0.999)
tfidf.matrix <- as.matrix(tdm.tfidf)

# Cosine distance matrix (useful for specific clustering algorithms)
library(proxy)
dist.matrix <- dist(tfidf.matrix, method = "cosine")
```

Appendix

```
truth.K=2
#Perform clustering
library(dbscan)
clustering.kmeans <- kmeans(tfidf.matrix, truth.K)
clustering.hierarchical <- hclust(dist.matrix, metho
d = "ward.D2")
clustering.dbscan <- hdbscan(dist.matrix, minPts
= 10)

library(cluster)
clusplot(as.matrix(dist.matrix),clustering.kmeans$cl
uster,color=T,shade=T,labels=2,lines=0)
plot(clustering.hierarchical)
rect.hclust(clustering.hierarchical,2)
plot(as.matrix(dist.matrix),col=clustering.dbscan$cl
uster+1L)

#Combine results
master.cluster <- clustering.kmeans$cluster
slave.hierarchical <- cutree(clustering.hierarchical,
k = truth.K)
slave.dbscan <- clustering.dbscan$cluster
```

```
#plotting results
library(colorspace)
points <- cmdscale(dist.matrix, k = 2)
palette <- diverge_hcl(truth.K) # Creating a color
palette, need library(colorspace)
#layout(matrix(1:3,ncol=1))

plot(points, main = 'K-Means clustering', col = a
s.factor(master.cluster),
     mai = c(0, 0, 0, 0), mar = c(0, 0, 0, 0),
     xaxt = 'n', yaxt = 'n', xlab = '', ylab = '')
plot(points, main = 'Hierarchical clustering', col
= as.factor(slave.hierarchical),
     mai = c(0, 0, 0, 0), mar = c(0, 0, 0, 0),
     xaxt = 'n', yaxt = 'n', xlab = '', ylab = '')
plot(points, main = 'Density-based clustering', c
ol = as.factor(slave.dbscan),
     mai = c(0, 0, 0, 0), mar = c(0, 0, 0, 0),
     xaxt = 'n', yaxt = 'n', xlab = '', ylab = '')

table(master.cluster)
table(slave.hierarchical)
table(slave.dbscan)
```

Appendix

```
#Elbow plot
#accumulator for cost results
cost_df <- data.frame()

#run kmeans for all clusters up to 100
for(i in 1:20){
  #Run kmeans for each level of i, allowing up to
100 iterations for convergence
  kmeans<- kmeans(x=tfidf.matrix, centers=i, iter.
max=100)

  #Combine cluster number and cost together, wri
te to df
  cost_df<- rbind(cost_df, cbind(i, kmeans$tot.withi
nss))
}
names(cost_df) <- c("cluster", "cost")

plot(cost_df$cluster, cost_df$cost)
lines(cost_df$cluster, cost_df$cost)
```