

# Summarization

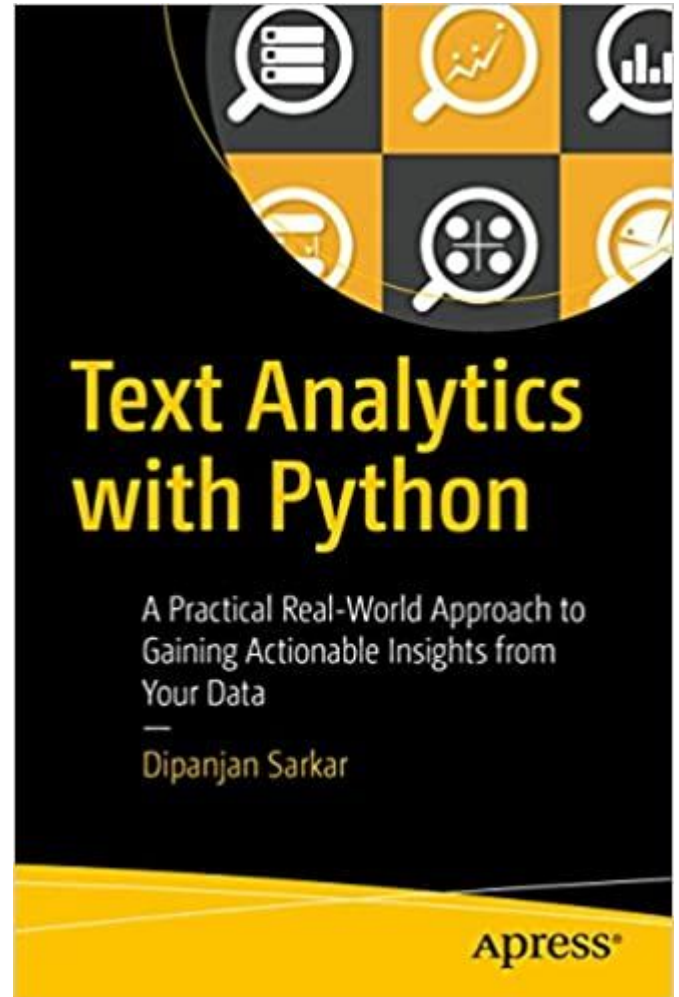
Topic Modeling & TextRank

# Outline

- Summarization approaches
- Applications of summarization
- Summarization using LSA and TextRank

# Reference

## Chapter 5. Text Summarization



# Approaches

- **Extractive summarization** means identifying important sections of the text and producing a subset of the sentences from the original text
- **Abstractive summarization** reproduces important material in a new way after interpretation and examination of the text using advanced natural language techniques to generate a new shorter text that conveys the most critical information from the original one
- Obviously, abstractive summarization is more advanced and closer to human-like interpretation. Though it has more potential, so far the more traditional methods have proved to yield better results

# Applications

- headlines (from around the world)
- outlines (notes for students)
- minutes (of a meeting)
- previews (of movies)
- synopses (soap opera listings)
- reviews (of a book, CD, movie, etc.)
- digests (TV guide)
- biography (resumes, obituaries)
- abridgments (Shakespeare for children)
- bulletins (weather forecasts/stock market reports)

# Methods

- Topic Modeling, e.g., LSA, LDiA
- TextRank

# Summarization using LSA

$k$  = # of topics,  $n$  = # of sentences of summary

1. When a document to be summarized is given, tokenize it into sentences
2. Vectorize the sentences
3. Build a term-document matrix, where a document corresponds to a sentence
4. Apply LSA to get  $U$ ,  $S$ ,  $V^t$
5. Get the sentence vectors from  $V$  ( $k$  rows)
6. Get the top  $k$  singular values from  $S$
7. Remove singular values that are less than half of the largest singular value (a heuristic)
8. Compute sentence weights per topic by multiplying each term sentence column from  $V$  squared with its corresponding singular value from  $S$  also squared
9. Compute the salience scores for each sentence by summing up the sentence weights across the topics and take the square root of the final score

# SVD (Singular Value Decomposition)

$$\begin{array}{c} A \\ \square \end{array} = \begin{array}{c} U \\ \square \end{array} \begin{array}{c} \Sigma \\ \square \end{array} \begin{array}{c} V^T \\ \square \end{array}$$

$$A = U\Sigma V^T$$

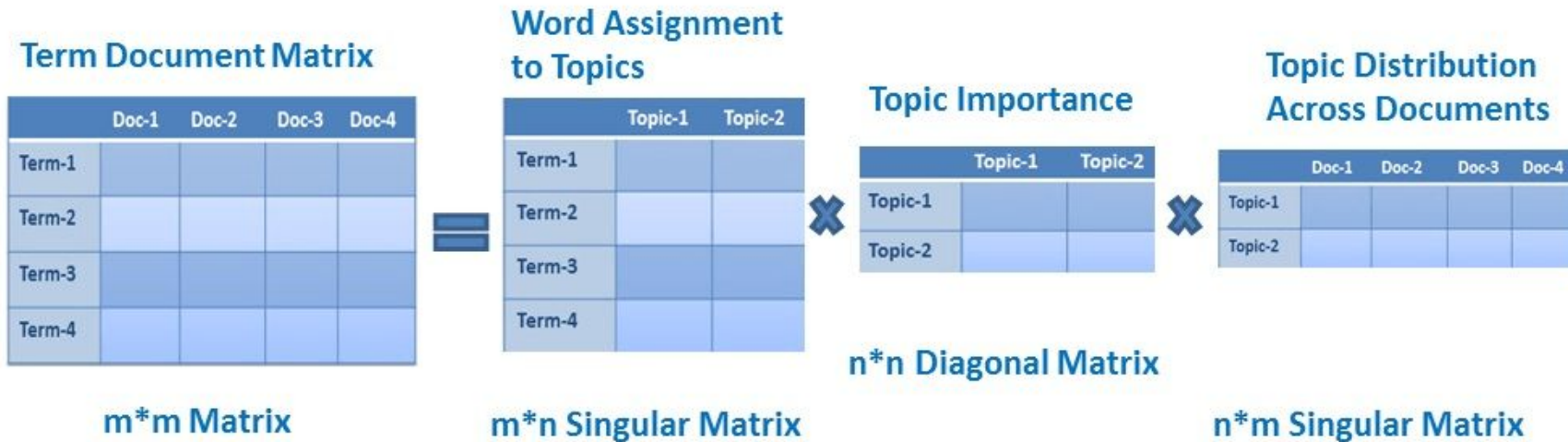
$U : m \times m$  직교행렬 ( $AA^T = U(\Sigma\Sigma^T)U^T$ )

$V : n \times n$  직교행렬 ( $A^TA = V(\Sigma^T\Sigma)V^T$ )

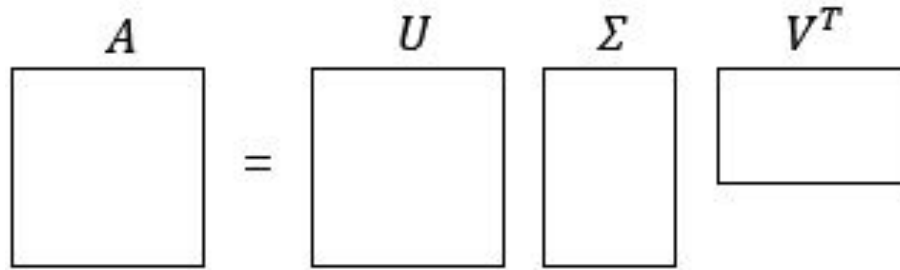
$\Sigma : m \times n$  직사각 대각행렬



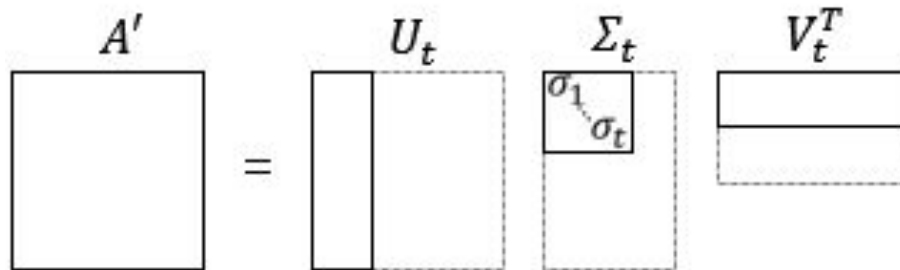
# SVD for Topic Modeling



# Truncated SVD (Singular Value Decomposition)

$$A = U \Sigma V^T$$




$$A' = U_t \Sigma_t V_t^T$$


$t = \# \text{ of topics}$

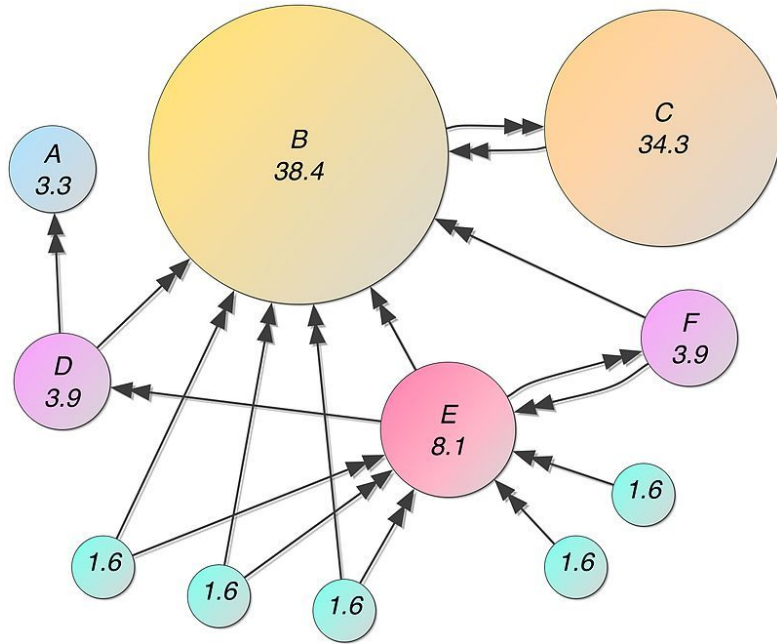
# Code

<https://github.com/Apress/text-analytics-w-python-2e/blob/master/Ch06%20-%20Text%20Summarization%20and%20Topic%20Models/Ch06e%20-%20Document%20Summarization.ipynb>

# TextRank

- TextRank algorithm **internally uses PageRank algorithm**, which is used by Google for ranking web sites and pages
- PageRank is a graph-based scoring or ranking algorithm, where pages are scored based on their importance. Web sites correspond to vertices and their links are edges in the graph
- Vertex importance is determined by the number of edges and the importance of the vertices that are connected to it

# PageRank (Brin and Page, 1998)



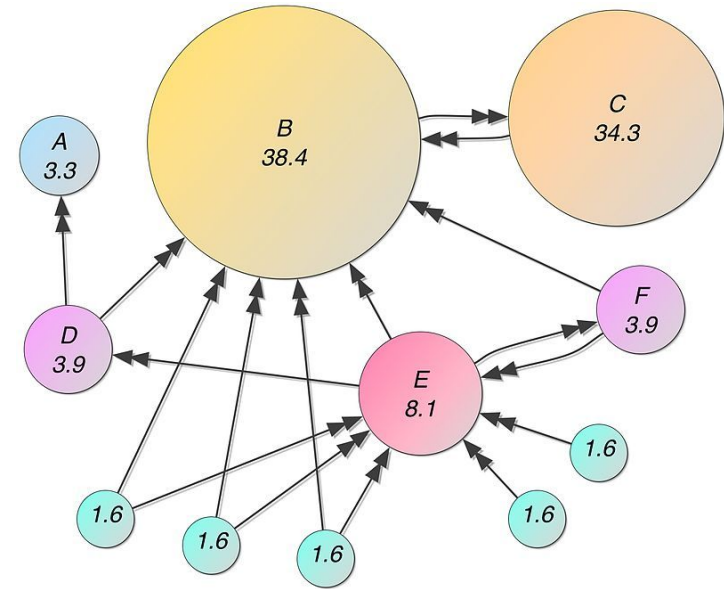
$$S(V_i) = (1 - d) + d * \sum_{j \in In(v_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

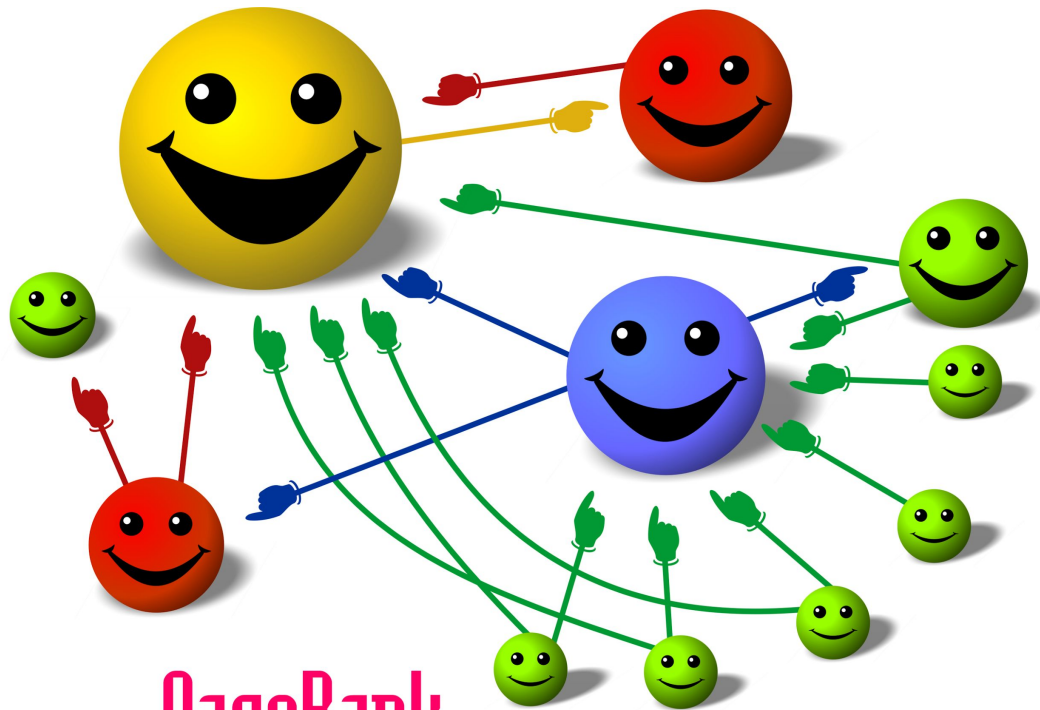
<https://en.wikipedia.org/wiki/PageRank>

# PageRank

$$S(V_i) = (1 - d) + d * \sum_{j \in In(v_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

- $S(V_i)$  - the weight of webpage  $i$
- $d$  - damping factor, in case of no outgoing links
- $In(V_i)$  - inbound links of  $i$ , which is a set
- $Out(V_j)$  - outgoing links of  $j$ , which is a set
- $|Out(V_j)|$  - the number of outbound links



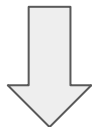


PageRank

$$S(V_i) = (1 - d) + d * \sum_{j \in In(v_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

# TextRank (EMNLP 2004)

$$S(V_i) = (1 - d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$



$$WS(V_i) = (1 - d) + d * \sum_{V_j \in In(V_i)} \frac{\sum_{V_k \in Out(V_j)} w_{jk}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

## TextRank: Bringing Order into Texts

Rada Mihalcea and Paul Tarau  
Department of Computer Science  
University of North Texas  
{rada,tarau}@cs.unt.edu

### Abstract

In this paper, we introduce TextRank – a graph-based ranking model for text processing, and show how this model can be successfully used in natural language applications. In particular, we propose two innovative unsupervised methods for keyword and sentence extraction, and show that the results obtained compare favorably with previously published results on established benchmarks.

### 1 Introduction

Graph-based ranking algorithms like Kleinberg's HITS algorithm (Kleinberg, 1999) or Google's PageRank (Brin and Page, 1998) have been successfully used in citation analysis, social networks, and the analysis of the link-structure of the World Wide Web. Arguably, these algorithms can be singled out as key elements of the paradigm-shift triggered in the field of Web search technology, by providing a Web page ranking mechanism that relies on the collective knowledge of Web architects rather than individual content analysis of Web pages. In short, a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information.

Applying a similar line of thinking to lexical or semantic graphs extracted from natural language documents, results in a graph-based ranking model that can be applied to a variety of natural language processing applications, where knowledge drawn from an entire text is used in making local ranking/selection decisions. Such text-oriented ranking methods can be applied to tasks ranging from automated extraction of keyphrases, to extractive summarization and word sense disambiguation (Mihalcea et al., 2004).

In this paper, we introduce the TextRank graph-based ranking model for graphs extracted from natural language texts. We investigate and evaluate the application of TextRank to two language processing tasks consisting of unsupervised keyword and sen-

tence extraction, and show that the results obtained with TextRank are competitive with state-of-the-art systems developed in these areas.

### 2 The TextRank Model

Graph-based ranking algorithms are essentially a way of deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph. The basic idea implemented by a graph-based ranking model is that of “voting” or “recommendation”. When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex. Moreover, the importance of the vertex casting the vote determines how important the vote itself is, and this information is also taken into account by the ranking model. Hence, the score associated with a vertex is determined based on the votes that are cast for it, and the score of the vertices casting these votes.

Formally, let  $G = (V, E)$  be a directed graph with the set of vertices  $V$  and set of edges  $E$ , where  $E$  is a subset of  $V \times V$ . For a given vertex  $V_i$ , let  $In(V_i)$  be the set of vertices that point to it (predecessors), and let  $Out(V_i)$  be the set of vertices that vertex  $V_i$  points to (successors). The score of a vertex  $V_i$  is defined as follows (Brin and Page, 1998):

$$S(V_i) = (1 - d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

where  $d$  is a damping factor that can be set between 0 and 1, which has the role of integrating into the model the probability of jumping from a given vertex to another random vertex in the graph. In the context of Web surfing, this graph-based ranking algorithm implements the “random surfer model”, where a user clicks on links at random with a probability  $d$ , and jumps to a completely new page with probability  $1 - d$ . The factor  $d$  is usually set to 0.85 (Brin and Page, 1998), and this is the value we are also using in our implementation.



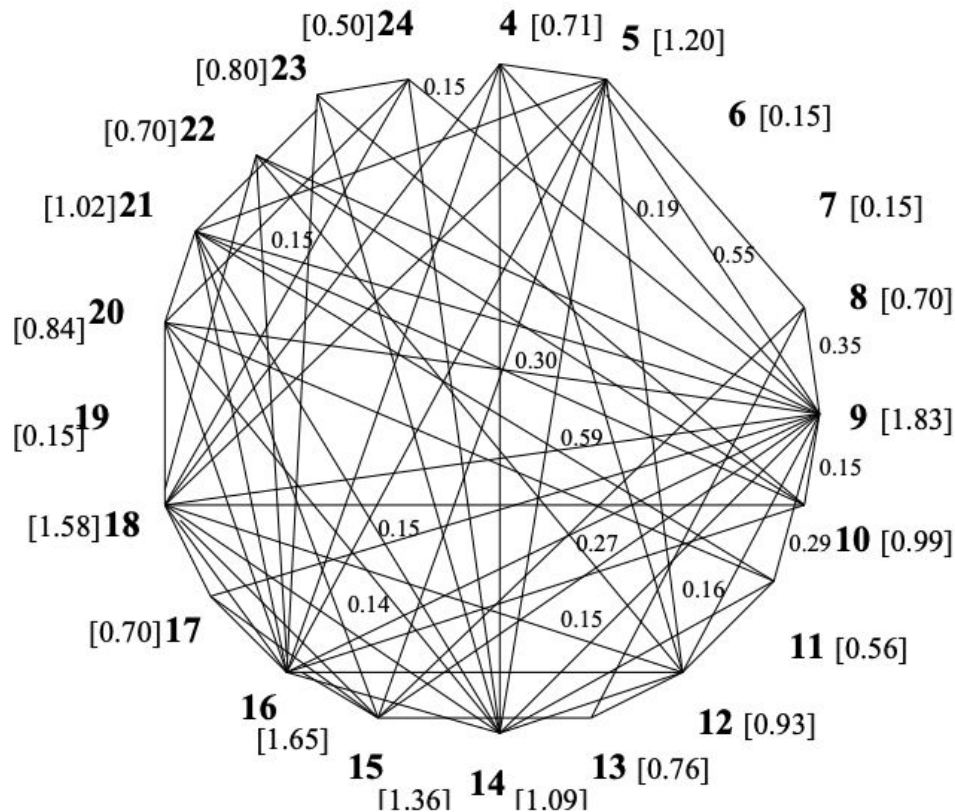
# TextRank

$k$  = # of sentences of summary

1. When a document to be summarized is given, tokenize it into sentences
2. Vectorize the sentences
3. Build a document-term matrix, where a document corresponds to a sentence
4. Compute a **document similarity matrix** by multiplying the matrix with its transpose (In original TextRank the weights of an edge between two sentences is the percentage of words appearing in both of them)
5. Use these documents (i.e. sentences) as the vertices and the similarities between each pair of documents as the weight and feed them to the PageRank algorithm
6. Get the score for each sentence
7. Rank the sentences based on score and return the top  $k$  sentences

# TextRank for Summarization

- 3: BC-Hurricane Gilbert, 09-11 339
- 4: BC-Hurricane Gilbert, 0348
- 5: Hurricane Gilbert heads toward Dominican Coast
- 6: By Ruddy Gonzalez
- 7: Associated Press Writer
- 8: Santo Domingo, Dominican Republic (AP)
- 9: Hurricane Gilbert Swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains, and high seas.
- 10: The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph.
- 11: "There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly after midnight Saturday.
- 12: Cabral said residents of the province of Barahona should closely follow Gilbert's movement.
- 13: An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo.
- 14: Tropical storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night.
- 15: The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo.
- 16: The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm.
- 17: The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.
- 18: Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds, and up to 12 feet to Puerto Rico's south coast.
- 19: There were no reports on casualties.
- 20: San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night.
- 21: On Saturday, Hurricane Florence was downgraded to a tropical storm, and its remnants pushed inland from the U.S. Gulf Coast.
- 22: Residents returned home, happy to find little damage from 90 mph winds and sheets of rain.
- 23: Florence, the sixth named storm of the 1988 Atlantic storm season, was the second hurricane.
- 24: The first, Debby, reached minimal hurricane strength briefly before hitting the Mexican coast last month.



# TextRank for Summarization

## **TextRank extractive summary**

Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo. The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm. Strong winds associated with Gilbert brought coastal flooding, strong southeast winds and up to 12 feet to Puerto Rico's south coast.

## **Manual abstract I**

Hurricane Gilbert is moving toward the Dominican Republic, where the residents of the south coast, especially the Barahona Province, have been alerted to prepare for heavy rains, and high wind and seas. Tropical storm Gilbert formed in the eastern Caribbean and became a hurricane on Saturday night. By 2 a.m. Sunday it was about 200 miles southeast of Santo Domingo and moving westward at 15 mph with winds of 75 mph. Flooding is expected in Puerto Rico and in the Virgin Islands. The second hurricane of the season, Florence, is now over the southern United States and downgraded to a tropical storm.

## **Manual abstract II**

Tropical storm Gilbert in the eastern Caribbean strengthened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday to be about 140 miles south of Puerto Rico and 200 miles southeast of Santo Domingo. It is moving westward at 15 mph with a broad area of cloudiness and heavy weather with sustained winds of 75 mph gusting to 92 mph. The Dominican Republic's Civil Defense alerted that country's heavily populated south coast and the National Weather Service in San Juan, Puerto Rico issued a flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.

## Keyword extraction using TextRank

The diagram illustrates a network of mathematical concepts. The nodes are arranged in a circular fashion, with various arcs connecting them. The nodes include: systems, compatibility, criteria, numbers, natural, upper, bounds, components, construction, minimal, sets, solutions, types, linear, system, diophantine, constraints, equations, nonstrict, strict, inequations, algorithms, and criteria.

linear constraints; linear diophantine equations; natural numbers; nonstrict  
inequations; strict inequations; upper bounds

linear constraints; linear diophantine equations; minimal generating sets; non-strict inequations; set of natural numbers; strict inequations; upper bounds

# TextRank example code

<https://github.com/Apress/text-analytics-w-python-2e/blob/master/Ch06%20-%20Text%20Summarization%20and%20Topic%20Models/Ch06e%20-%20Document%20Summarization.ipynb>

# More reference

<https://rare-technologies.com/text-summarization-in-python-extractive-vs-abstractive-techniques-revisited/>

# Summary

- LSA for summarization computes the sentence importance by multiplying its topic singularity value and its corresponding value
- TextRank algorithm computes the sentence importance based on its similarities with the other sentences
- Similarity between two words/sentences is associated with their distance