DATA SCIENCE CONSULTING
Session 3

February 8th, 2021

Agenda

1. **Business case**
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Business case: what for?

The business case aims to **measure the benefits** a client can achieve by **following through a transformation journey**. It is an **essential deliverable** of a majority of our projects.

Strategic Scoping

Financial Analysis

Benefits Tracking

Bottom-Up Business Case

Top down Business Case

Composition of a business case

Through our workshops, we have started developing the project's B-case.

	Strategic Scoping	Financial analysis	Top-down Business Case	Bottom-Up Business Case	Benefits Tracking (RUN)
Purpose	▪ Understand the business model ▪ Develop hypothesis	▪ Quantify competitive gaps ▪ Identify focus for stream diagnostics	▪ Select costs and benefits drivers ▪ Discuss and validate assumptions ▪ Quantify preliminary volume of benefits	▪ Identify, quantify and validate financial benefits ▪ Summarize and present results of the bottom-up ▪ Business case	▪ Track realisation of identified benefits during the implementation phase
Data Collection Techniques	▪ Interviews ▪ Workshops ▪ Company and competition analysis	▪ Financial reporting ▪ KPI analysis ▪ Analysis of return on investment (ROI) ▪ Benchmarking	▪ Estimate the effect on the profit and loss (P&L) ▪ Calculate the cash flow for the proposed saving ▪ Calculate the projects ROI	▪ Develop a Benefits Logic ▪ Present Benefit opportunities ▪ Selection options prioritize and define phasing model	▪ Define and track KPIs for benefits
What we have done so far	***Scraping***	***Definition of KPIs***	**Today's focus**	***Suggestions based on analysis of scrapped data & other external sources***	***To be implemented in run***

Top-down business case (2/3)

Profits = Revenues - Costs

Revenues

- What are the different types of revenues generated by a restaurant?
- Could there even be revenues that go beyond the scope of the restaurant?
- Are there any patterns/seasonality in the revenues?

Costs

- What are the different types of costs associated with a restaurant? How could they be classified ?
- How could we estimate the cost of goods sold?
- How would these costs be shared with the hotel and potentially optimized?

Top-down business case (3/3)

Profits = Revenues - Costs

Revenues

Revenues = number of clients x individual spending

Assumption to calculate the number of clients

- XX% come for drink
- The capacity is 70 people with a potential XX% occupancy rate?
- One set of person would stay for XX hours
- The restaurant is opened from XX PM to XX PM

Assumption to calculate the amount spent

- On average, a person coming to eat would consume for XX of food and XX of drink
- On average, a person coming to drink would consume for XX of food and XX of drink

Costs

Total costs = Fixed costs + variable costs

Fixed costs

- The costs to rent the locations:
- Management / overhead costs including advertisement
- Electricity, water, other fixed resources costs:
- Costs to do the place rebranding (it could potentially be amortized)

Variable costs

- Cost of drinks and food: could it be a percentage of the expenditure per client?
- Cost of workforce: relation with number of clients?

Top-down business case(3/3)

Profits = Revenues - Costs

Revenues

- The two leverages on revenues are to either have more customers or have them spend more at every order
  - How to reach more customers? Should hotel resources be leveraged or even external sources?
  - Price: how could we justify a price increase? How would it increase the number of customers?
- How would revenue optimization impact other aspects of the hotel (costs, hotel brand image,…)?

Costs

How to reduce fixed and variable costs

Fixed costs

- Can we reduce overhead costs through XXX?

Variable

- How could we optimize the purchase of food and drinks? What would be the impact on quality?
- How could we optimize / reduce the workforce?

Brainstorm

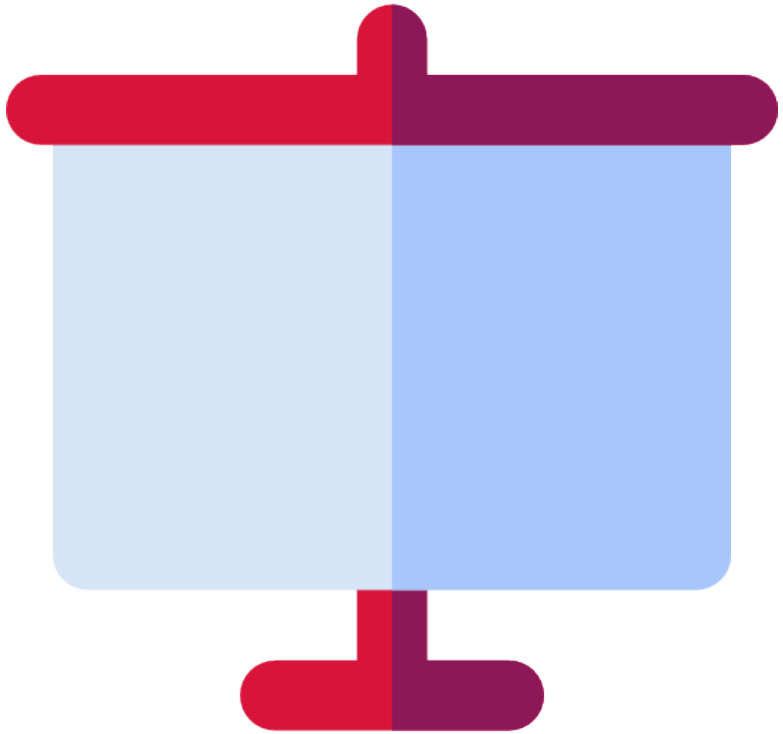
20’

Start to create your own quantitative business case !

Restitution

5’

Which information did you get ?



Agenda

1. Business case
2. **Governance bodies**
3. Back to TF-IDF approach
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Focus on the Governance

Governance body	Duration & frequency	Content
Sponsors meeting	▪ 1 hour ▪ Once a month	▪ Review of overall project results ▪ Validations of : – Strategic orientations – Cross-segment arbitrations and structuring decisions
Steering committee	▪ 1 hour ▪ Once a month	▪ Review of overall project results (advancements, added value, costs…) ▪ Breaking points and alerts ▪ Decision making or submission of arbitrations to the Sponsors
Project committee	▪ 1 hour ▪ Once a week	▪ Operational alignment on achievements and on-going actions ▪ Identifications of attention points
Core Team stand-up	▪ 15 minutes ▪ Each day	▪ Sharing of what has been done the day before and what is foreseen this day ▪ Solving of simple problems
Kick off meeting	▪ 1 hour ▪ Once	▪ Set project strategic objectives ▪ Define scope ▪ Define stakeholders, governance & macroplanning

Steering Committee

Participants

- **Project leaders :** your client

Objectives

- **Review overall project results**
  - Overall status
  - Latest Results
  - Foreseen next steps
- **Suggestions for content**
  - Latest version of NLP results
  - Quantified objectives of **expected revenue** after rebranding
  - Prepare slides

Format

- **On a regular basis, all along the project progress**
  - On average once a week
- **Short and efficient – focused on results & key questions**
  - Duration ~ hour

Steering Committee

Ex: implement a Data Platform

Data Platform industrialization (see slide 5)

Collaborative work on Platform architecture (see slide 4)

Share shortlisted business use cases (see slide 10)

FOR INFORMATION

Buidling Target Operating Model
(ref TOM slide 25)

FOR DECISION

Validate the choice of the cloud provider further to the RFP process

Validate the planning of the Platform implementation, both Pilot and target platform and operational impacts on the Data Department

Confirm positions to open within the Data Department

Confirm the agile mode to build projects within the Data Platform

For next week

Please draft and share with us your Steering Committee vision with main topics:

Key decisions since the last SteerCo…

Impacted on :

Key achievements…

In a context …

Where decisions should be taken on …:

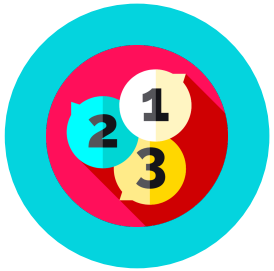
Agenda

1. Business case
2. Governance bodies
3. **Back to TF-IDF approach**
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Data pipeline

Data Collection

Data Cleaning



Word Embedding

Topic Extraction

Sentiment Analysis

Agenda

1. Business case
2. Governance bodies
3. **Back to TF-IDF approach**
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Back to the TF-IDF approach

Raw data

Document ID	Textual description
1	"Data science is fun"
2	"Artificial intelligence is the future"
3	"Business and artificial intelligence combination is the key"

Cleaning

Cleaned data

Document ID	Cleaned text
1	["data", "science" , "fun"]
2	["artificial", "intelligence" , "future"]
3	["business", "artificial", "intelligence", "combination", "key"]

Vocabulary extraction

Dictionary

["data", "science" , "fun", "artificial", "intelligence" , "future", "business", "combination", "key"]

$$TF_{ij} = \frac{w_{ij}}{|d_j|} \qquad IDF_i = \log\frac{N}{df_i} \qquad w_{ij} = TF_{ij}.IDF_i$$

TF

Word / Document	data	science	fun	artificial	intelligence	future	business	combination	key
1	0.33	0.33	0.33	0	0	0	0	0	0
2	0	0	0	0.33	0.33	0.33	0	0	0
3	0	0	0	0.2	0.2	0	0.2	0.2	0.2

IDF

Word / Document	data	science	fun	artificial	intelligence	future	business	combination	key
1	0,48	0,48	0,48	0,18	0,18	0,48	0,48	0,48	0,48
2	0,48	0,48	0,48	0,18	0,18	0,48	0,48	0,48	0,48
3	0,48	0,48	0,48	0,18	0,18	0,48	0,48	0,48	0,48

TF-IDF

Word / Document	data	science	fun	artificial	intelligence	future	business	combination	key
1	0.16	0.16	0.16	0,00	0,00	0,00	0,00	0,00	0,00
2	0,00	0,00	0,00	0,06	0,06	0.16	0,00	0,00	0,00
3	0,00	0,00	0,00	0,04	0,04	0,00	0,1	0,1	0,1

LIMITS

- ✓ Syntax
- ✓ Semantics (meaning and relationship)
- ✓ Synonyms
- ✓ Sparsity

Agenda

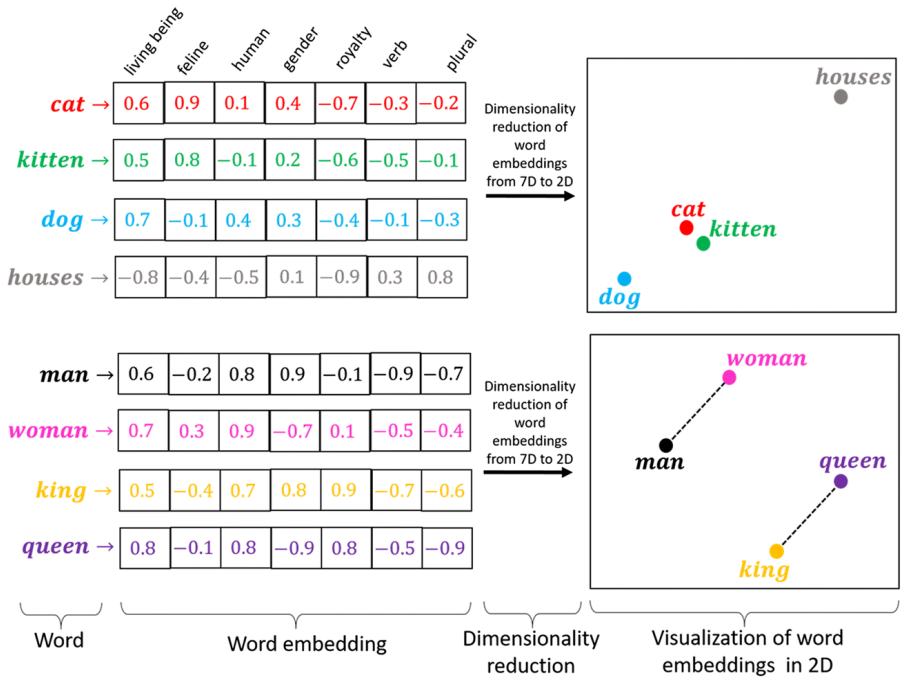
1. Profits' calculation
2. Governance bodies
3. Back to TF-IDF approach
4. **Word embedding definition**
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Word embedding definition[1]

Word embedding is a type of word representation that allows words with similar meaning to have a similar representation.

Before word embedding

Man != Woman



After word embedding

Man ~ Woman

ADVANTAGES

- ✓ **Semantics (meaning and relationship) embedded**
- ✓ **Dense and low-dimensional vectors**

[1] GOLDBERG, Yoav. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 2017, vol. 10, no 1, p. 1-309.

Word embedding definition

"Data science is fun …"

["data", "science" , "fun"]

1	0	0	…
0	1	0	
0	0	1	
⋮			

Main principle of functioning

TRAINING

Transformation matrix
(Latent parameters)

BUILD

Embedding model

Using on new example (word)

"man"

⋮
1
0
⋮

Embedding model

word	weight
woman	0.76
boy	0.68
girl	0.59
⋮	⋮

Agenda

1. Business case
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. **Word embedding approaches**
    A. **Latent Semantic Indexing (LSI) technique**
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Latent Semantic Indexing (LSI) technique (1/7)

Latent Semantic Analysis/Indexing (LSA/LSI) is a mathematical method for modeling the meaning of words by analysing a corpus of texts

- Map each **document** into some **concepts**
- Map each **term** into some **concepts**

Concepts are defined as a set of terms, with corresponding weights

For example, Sport concept :
« ball » (0.8)
« player » (0.6)
« run » (0.5)

Term-concept matrix

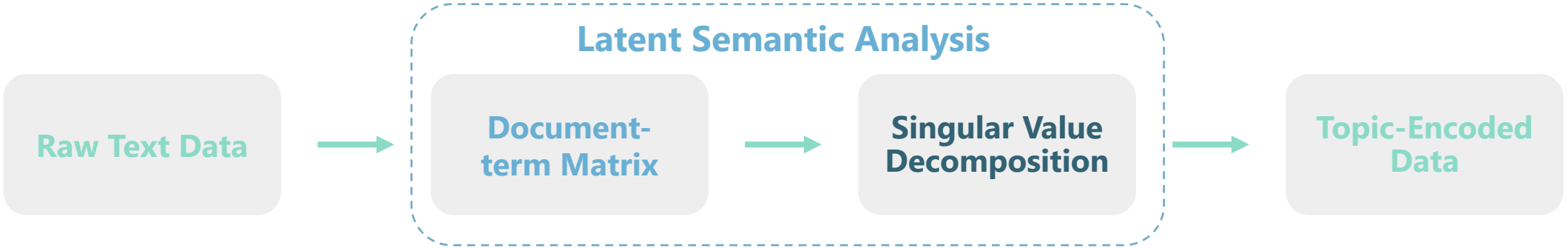
	Sport concept	Cooking concept
ball	0.8	
player	0.6	
run	0.5	
cake		0.9
oven		0.4

Document-concept matrix

	Sport concept	Cooking concept
Doc 1	0.7	
Doc 2	0.6	
Doc 3		0.5
Doc 4		0.2

[2] Thomas Landauer and Susan Dumais (2008) Latent semantic analysis. Scholarpedia, 3(11):4356

Latent Semantic Indexing (LSI) technique (2/7)



Document-term Matrix

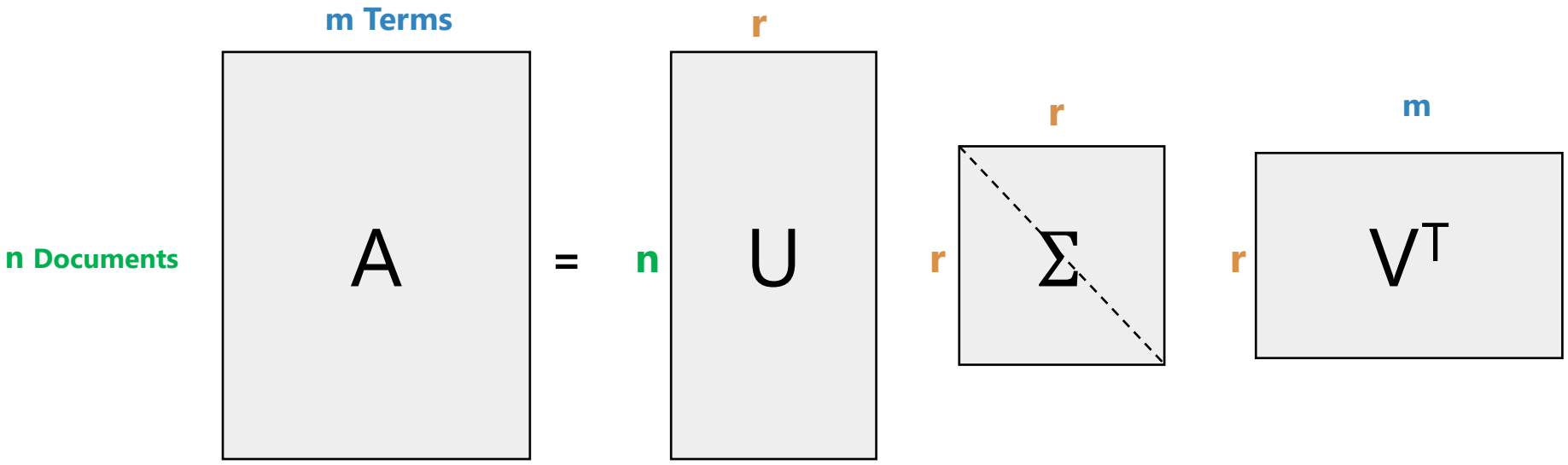
- Get a large collection of texts, representative of human language
- Build a matrix with documents as row and terms as columns (TF, TF-IDF…)

Singular Value Decomposition

- Document-term matrix is **decomposed** into product of matrices using SVD method
- **Dimensionality reduction** can be applied, keeping the k largest singular values and their associated vectors (LSI-space)

Latent Semantic Indexing (LSI) technique (3/7)

Singular Value Decomposition



$$\boldsymbol{A}_{[n \times m]} = \mathbf{U}_{[n \times r]} \boldsymbol{\Sigma}_{[r \times r]} (\mathbf{V}_{[m \times r]})^T$$

Latent Semantic Indexing (LSI) technique (4/7)

Similarity in LSI Space

Term-Term similarity

$$A^tA = (U\Sigma V^T)^T(U\Sigma V^T)$$

$$A^tA = (V^T\Sigma^T U^{TT})(U\Sigma V^T)$$

$$A^tA = (V^T\Sigma^T)(\Sigma V^T)$$

$$A^tA = V\Sigma^2 V^T$$

V are the eigenvectors of the covariance matrix A^tA.

Document-Document similarity

$$AA^t = (U\Sigma V^T)(U\Sigma V^T)^T$$

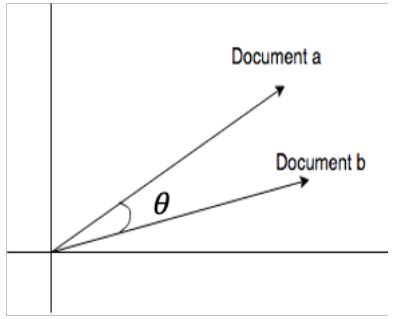
$$AA^t = (U\Sigma V^T)(V^{TT}\Sigma^T U^T)$$

$$AA^t = (U\Sigma)(\Sigma^T U^T)$$

$$AA^t = U\Sigma^2 U^T$$

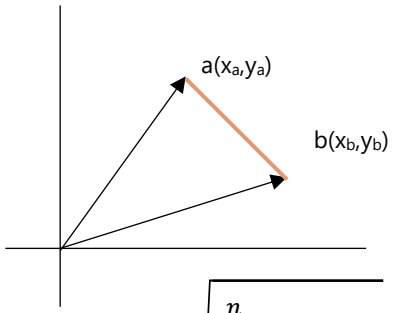
U are the eigenvectors of the Gram matrix AA^t.

Cosine similarity



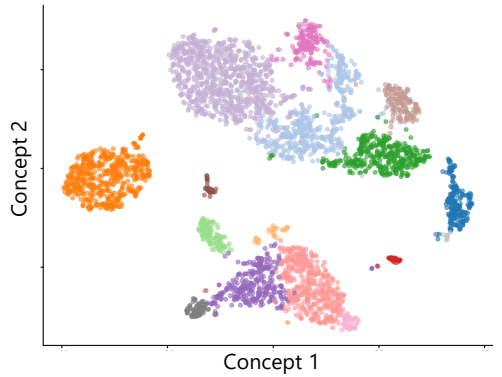
$$sim(a,b) = \cos(\theta) = \frac{\vec{a}.\vec{b}}{||\vec{a}||.||\vec{b}||}$$

Euclidian norm



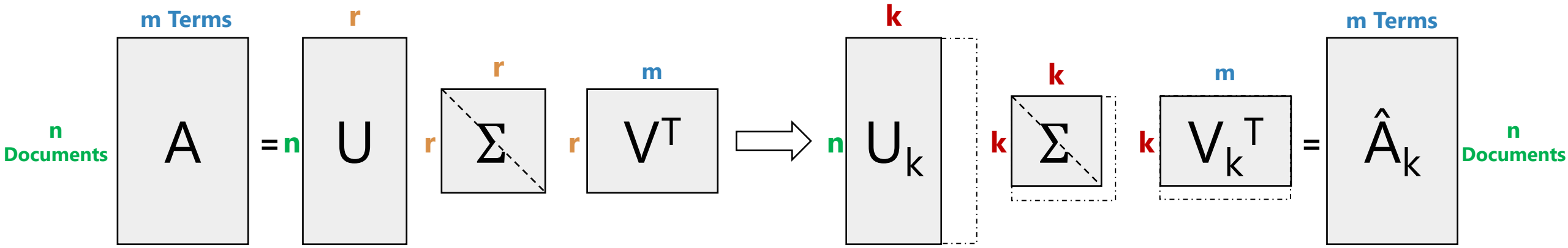
$$d(a,b) = \sqrt{\sum_{i=1}^{n}(ai - bi)}$$

Clustering techniques can also be used with the new LSI-space vectors



Latent Semantic Indexing (LSI) technique (5/7)

SVD enables dimensionality reduction



We perform a SVD
U and V are orthogonal matrices
Σ is a diagonal matrix

The matrix $\hat{A}_k$ is obtained by keeping the first k columns of U and V, and the k largest elements of Σ

To fix an optimal value of "k" we can rely on the construction error: error(A, $\hat{A}$)

Latent Semantic Indexing (LSI) technique (6/7)

Advantages

- Powerful and generalizable tool
- Enables dimensionality reduction in an easily interpretable space
- Gives a good way to compute distance or similarity between documents/terms
- Helps for various NLP tasks : search and retrieval, classification, filtering

Weaknesses

- Interpretable meaning of LSI transformation can be complex sometimes even though the mathematical part is valid[2]
- LSI is very sensitive to new type of text that are widely used today[2] (contracted words, slang…)
- LSI always preserves linear regularities among words[3], hard to deal with synonyms

[3] Mikolov, Yih, Zweig, 2013, Linguistic Regularities in Continuous Space Word Representations

Latent Semantic Indexing (LSI) technique (7/7)

30’

Implement LSI in python to extract hidden features from your text data and perform dimensionality reduction

Break

10’

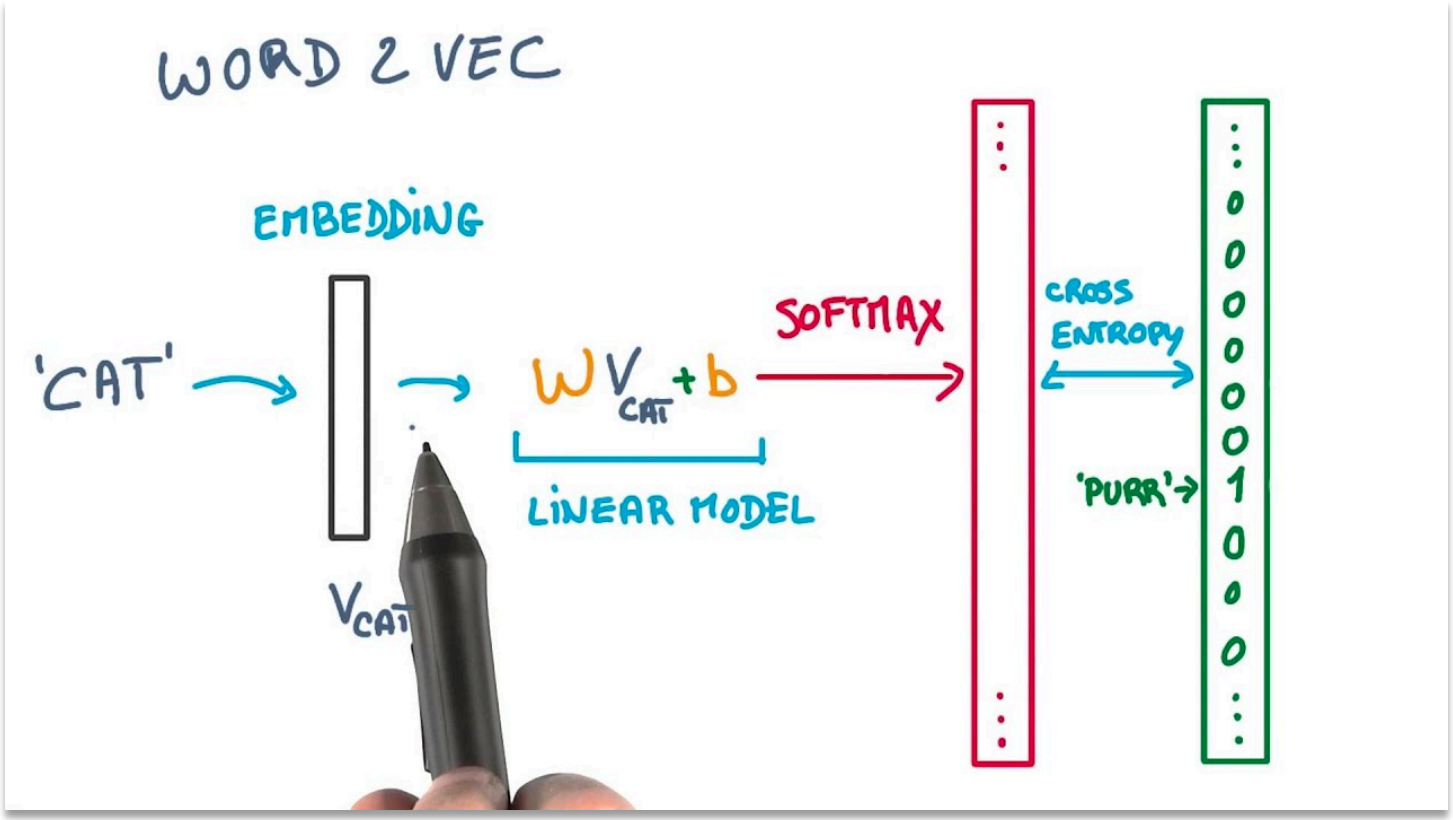
Feel free to help yourself ! See you at 16.30 !

Agenda

1. Business case
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. **Word embedding approaches**
   A. Latent Semantic Indexing (LSI) technique
   B. An advanced embedding method: Word2Vec
   C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

An advanced embedding method: Word2Vec (1/10)

Word2Vec is a representation of document vocabulary in which words with similar context occupy close spatial positions[4]

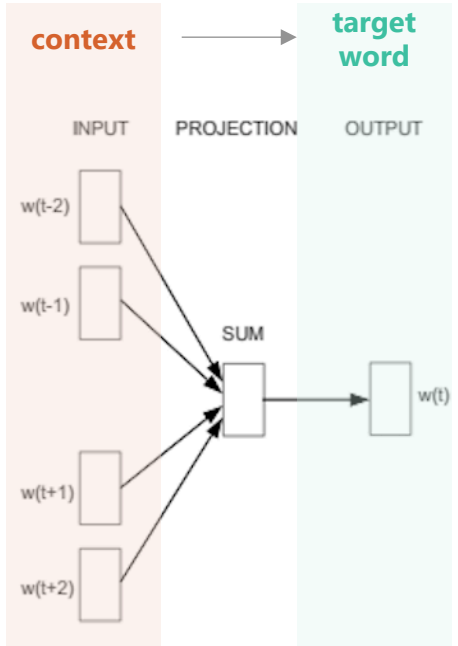


[4] Dhruvil Karani 2018, Introduction to word embedding and word2vec

An advanced embedding method: Word2Vec (3/10)

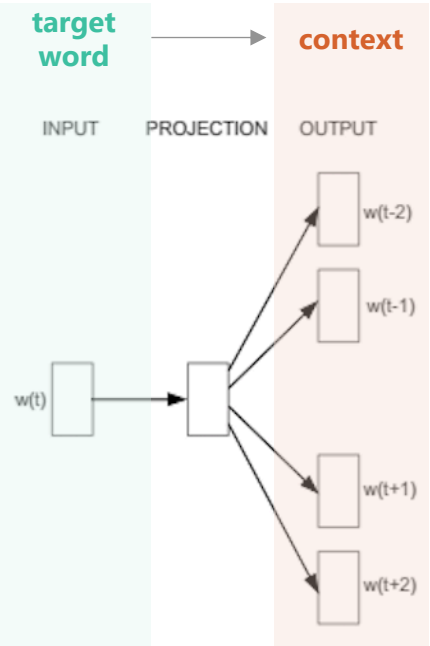
Continuous Bag of words (CBOW)

Takes the **context** of each word as the input and tries to predict the word corresponding to the context being the **target**



Skipgram

Takes the **target** word and tries to predict the **context** words of that target word and produce representations



Corpus example: "**The cat jumped over the puddle**"
{"The", "cat", "over", "the", "puddle"} => **context words**
{"jumped" } => **target word**

[6] Mikolov, Chen, Corrado, Dean 2013a, Efficient estimation of Words Representations in Vector Space

An advanced embedding method: Word2Vec (4/10)

Zoom on CBOW: One-word context[7]

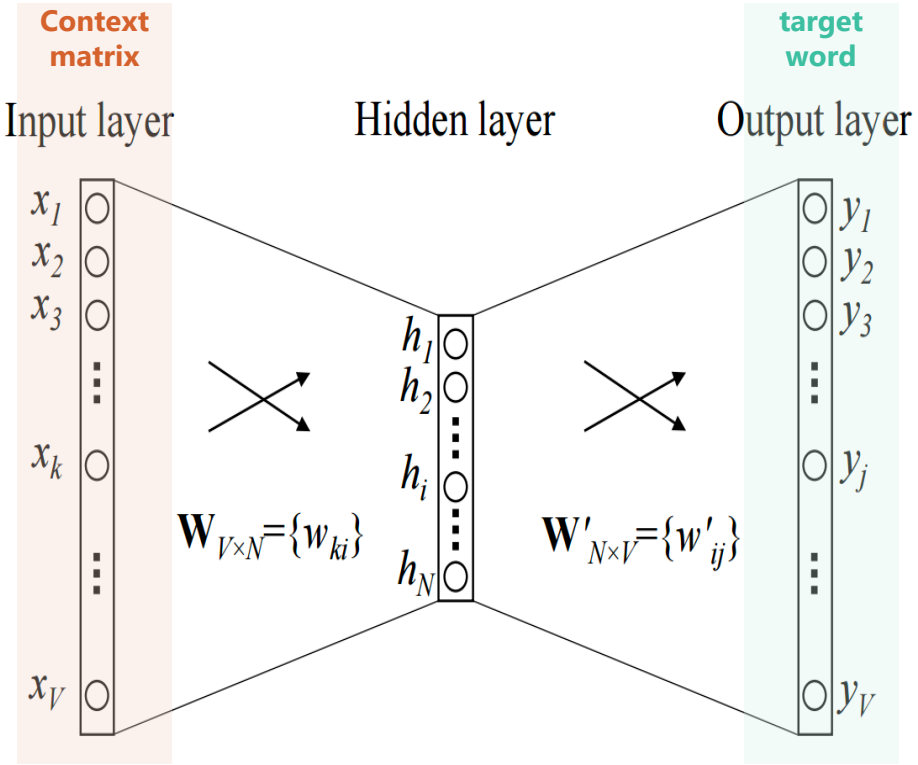
$$\boldsymbol{h = WTx = W^{T}_{(k,.)}}$$

$\boldsymbol{u_j = v'^{T}_{w_j} h}$ where $\boldsymbol{v'_{w_j}}$ is the j-th column of W'

$$\boldsymbol{y_j = \frac{exp(uj)}{\sum_{j'=1}^{V} exp(u_{j'})}}$$

Such as:

- **x** : one-hot encoded vector of a word
- **W** : weight matrix of size $V * N$
- **W'** : weight matrix of size $N * V$
- **V** : size of the vocabulary
- **N** : hidden layer size



Corpus example: "The **cat** **jumped** over the puddle"
{"The", "**cat**", "over", "the", "puddle"} => [0, **1**, 0, 0, 0, 0] => **context matrix**
{"jumped" } => **target word**

[7] Xin Rong 2016, Word2Vec Parameter Learning Explained

An advanced embedding method: Word2Vec (5/10)

Zoom of CBOW: Multi-word context[7]

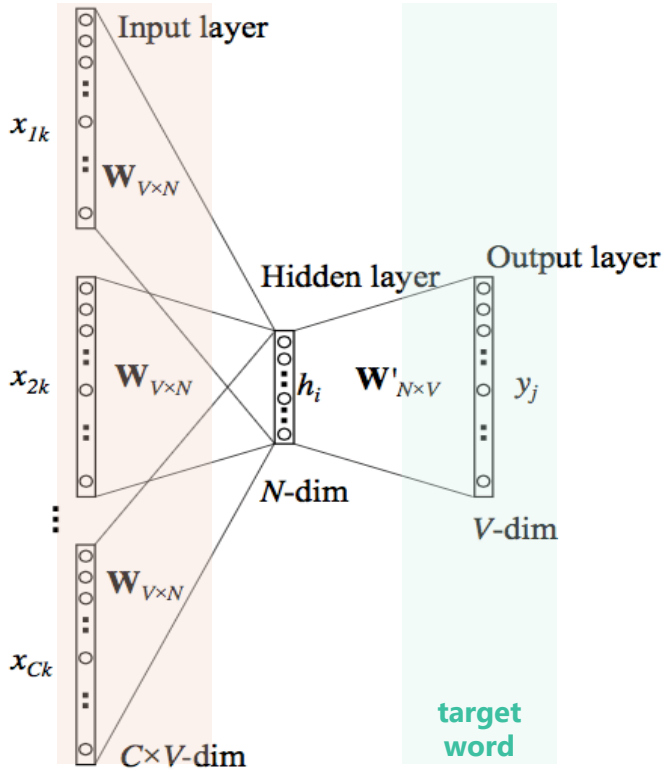
$$\boldsymbol{h} = \frac{\mathbf{1}}{\boldsymbol{C}}\boldsymbol{W}^{\boldsymbol{T}}(\boldsymbol{x}_1 + \boldsymbol{x}_2 + \cdots + \boldsymbol{xC})$$

$$\boldsymbol{h} = \frac{\mathbf{1}}{\boldsymbol{C}}(\boldsymbol{vw}_1 + \boldsymbol{vw}_2 + \cdots + \boldsymbol{vwc})^{\mathbf{T}}$$

Such as:

- **C** : number of the context word
- **V** : size of the vocabulary
- **N** : hidden layer size

Score function and **softmax** output as previously



Corpus example: "The **cat** **jumped** over the **puddle**"
{"The", "**cat**", "over", "the", "**puddle**"} => [0, **1**, 0, 0, 0, 0] & [0, 0, 0, 0, 0, **1**] => **context matrix** **or input matrix**
{"jumped" } => **target word**

[7] Xin Rong 2016, Word2Vec Parameter Learning Explained

An advanced embedding method: Word2Vec (6/10)

Zoom on Skipgram[7]

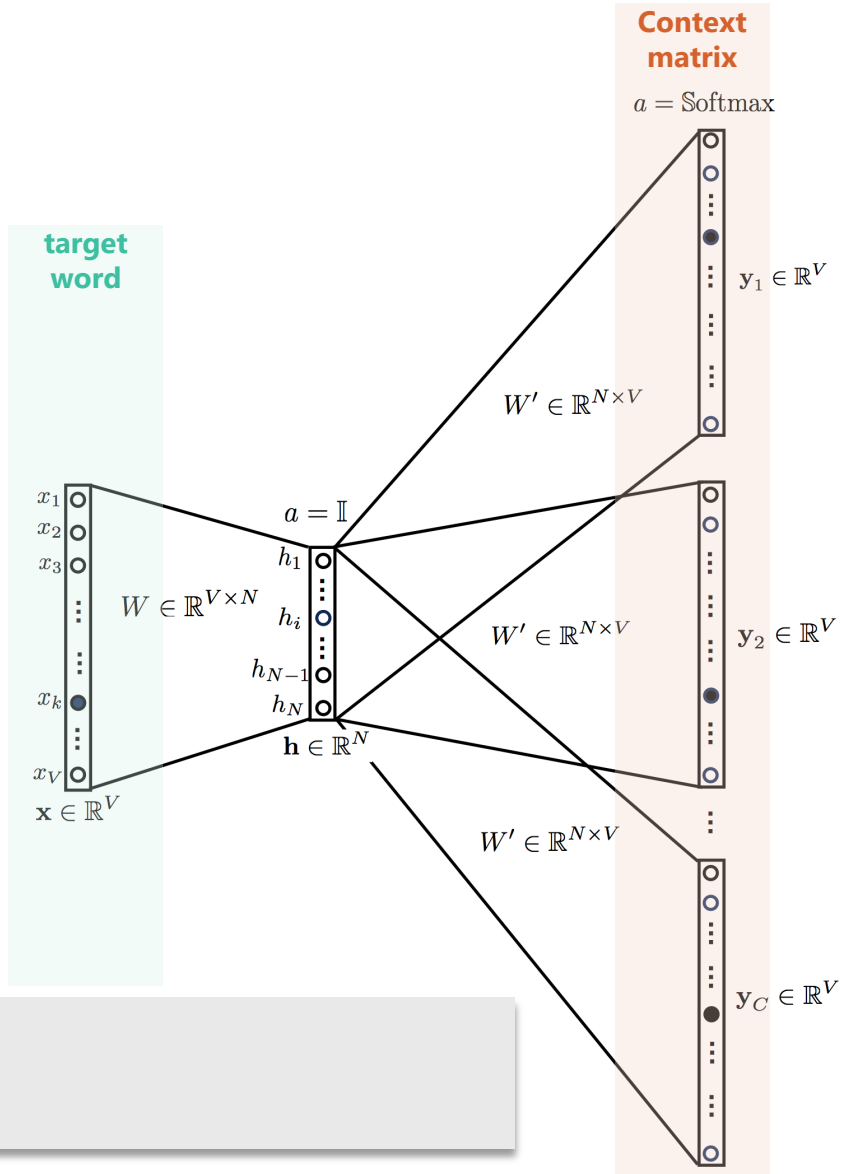
$$\boldsymbol{h = WTx = W^{T}_{(k,.)}}$$

$\boldsymbol{u_{c,j} = v'^{T}_{W_j} h}$ for c in 1,..., C where $\boldsymbol{v'_{W_j}}$ is the j-th column of W'

$\boldsymbol{u_{c,j} = u_j}$ because the output layer share the same weight matrix

$$\boldsymbol{y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{j'})}}$$

- **V** : size of the vocabulary
- **N** : hidden layer size



Corpus example: "The **cat** **jumped** over the **puddle**"
{"The", "**cat**", "over", "the", "**puddle**"} => [0, **1**, 0, 0, 0, 0] & [0, 0, 0, 0, 0, **1**] => **context matrix**
{"jumped" } => **target word** **or input matrix**

[7] Xin Rong 2016, Word2Vec Parameter Learning Explained

An advanced embedding method: Word2Vec (7/10)

Step-by-step example[8]

natural language processing and machine learning is fun and exciting

Contexts and targets

#1	natural Xĸ	language Y(c=1)	processing Y(c=2)	and	machine	learning	is	fun	and	exciting	#1
#2	natural Y(c=1)	language Xĸ	processing Y(c=2)	and Y(c=3)	machine	learning	is	fun	and	exciting	#2
#3	natural Y(c=1)	language Y(c=2)	processing Xĸ	and Y(c=3)	machine Y(c=4)	learning	is	fun	and	exciting	#3
#4	natural	language Y(c=1)	processing Y(c=2)	and Xĸ	machine Y(c=3)	learning Y(c=4)	is	fun	and	exciting	#4
#5	natural	language	processing Y(c=1)	and Y(c=2)	machine Xĸ	learning Y(c=3)	is Y(c=4)	fun	and	exciting	#5
#6	natural	language	processing	and Y(c=1)	machine Y(c=2)	learning Xĸ	is Y(c=3)	fun Y(c=4)	and	exciting	#6
#7	natural	language	processing	and	machine Y(c=1)	learning Y(c=2)	is Xĸ	fun Y(c=3)	and Y(c=4)	exciting	#7
#8	natural	language	processing	and	machine	learning Y(c=1)	is Y(c=2)	fun Xĸ	and Y(c=3)	exciting Y(c=4)	#8
#9	natural	language	processing	and	machine	learning	is Y(c=1)	fun Y(c=2)	and Xĸ	exciting Y(c=3)	#9
#10	natural	language	processing	and	machine	learning	is	fun Y(c=1)	and Y(c=2)	exciting Xĸ	#10

What is the number of the Window=?
what is the total number of word in the corpus=?

One-hot encoding pass

target *Context 1* *Context 2*

#	Token	#1			#2				#3					#4					#5				
0	natural	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	language	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
2	processing	0	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0
3	and	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0
4	machine	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0
5	learning	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
6	is	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
7	fun	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	exciting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		Xĸ	Y(c=1)	Y(c=2)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)

#	Token	#6					#7					#8					#9				#10		
0	natural	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	language	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	processing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	and	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1
4	machine	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	learning	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	is	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
7	fun	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	1	0
8	exciting	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0
		Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Y(c=4)	Xĸ	Y(c=1)	Y(c=2)	Y(c=3)	Xĸ	Y(c=1)	Y(c=2)

[8] Derek Chai 2018, An implementation guide to Word2Vec using Numpy and Google Sheets

An advanced embedding method: Word2Vec (8/10)

Step-by-step example[8]

natural language processing and machine learning is fun and exciting

Forward propagation

Parameters - Embedding size: 10
Random initialisation (Range): -1 to 1

Calculate Hidden Layer

#	Token	Input - w_t
0	natural	1
1	language	0
2	processing	0
3	and	0
4	machine	0
5	learning	0
6	is	0
7	fun	0
8	exciting	0

1 x 9

np.dot

Weight 1 - W1 (9 x 10)

0.236	-0.962	0.686	0.785	-0.454	-0.833	-0.744	0.677	-0.427	-0.066
-0.907	0.894	0.225	0.673	-0.579	-0.428	0.685	0.973	-0.070	-0.811
-0.576	0.658	-0.582	-0.112	0.662	0.051	-0.401	-0.921	-0.158	0.529
0.517	0.436	0.092	-0.835	-0.444	-0.905	0.879	0.303	0.332	-0.275
0.859	-0.890	0.651	0.185	-0.511	-0.456	0.377	-0.274	0.182	-0.237
0.368	-0.867	-0.301	-0.222	0.630	0.808	0.088	-0.902	-0.450	-0.408
0.728	0.277	0.439	0.138	-0.943	-0.409	0.687	-0.215	-0.807	0.612
0.593	-0.699	0.020	0.142	-0.638	-0.633	0.344	0.868	0.913	0.429
0.447	-0.810	-0.061	-0.495	0.794	-0.064	-0.817	-0.408	-0.286	0.149

=

Hidden Layer - h (1 x 10)

0.236
-0.962
0.686
0.785
-0.454
-0.833
-0.744
0.677
-0.427
-0.066

Calculate y_pred

Hidden Layer - h (1 x 10)

0.236
-0.962
0.686
0.785
-0.454
-0.833
-0.744
0.677
-0.427
-0.066

np.dot

Weight 2 - W2 (10 x 9)

-0.868	-0.406	-0.288	-0.016	-0.560	0.179	0.099	0.438	-0.551
-0.395	0.890	0.685	-0.329	0.218	-0.852	-0.919	0.665	0.968
-0.128	0.685	-0.828	0.709	-0.420	0.057	-0.212	0.728	-0.690
0.881	0.238	0.018	0.622	0.936	-0.442	0.936	0.586	-0.020
-0.478	0.240	0.820	-0.731	0.260	-0.989	-0.626	0.796	-0.599
0.679	0.721	-0.111	0.083	-0.738	0.227	0.560	0.929	0.017
-0.690	0.907	0.464	-0.022	-0.005	-0.004	-0.425	0.299	0.757
-0.054	0.397	-0.017	-0.563	-0.551	0.465	-0.596	-0.413	-0.395
-0.838	0.053	-0.160	-0.164	-0.671	0.140	-0.149	0.708	0.425
0.096	-0.995	-0.313	0.881	-0.402	-0.631	-0.660	0.184	0.487

=

Output Layer (1 x 9)

1.258
-1.369
-1.828
1.196
0.545
1.113
1.333
-1.528
-2.335

Softmax - y_pred (1 x 9)

0.218
0.016
0.010
0.205
0.107
0.189
0.235
0.013
0.006

probabilities

Error calculation

y_pred Softmax	w_c = 1	Token	Diff	y_pred Softmax	w_c = 2	Token	Diff	EI Sum of Diff
0.218	0	natural	0.218	0.218	0	natural	0.218	0.436
0.016	1	language	-0.984	0.016	0	language	0.016	-0.968
0.010	0	processing	0.010	0.010	1	processing	-0.990	-0.980
0.205	0	and	0.205	0.205	0	and	0.205	0.411
0.107	0	machine	0.107	0.107	0	machine	0.107	0.214
0.189	0	learning	0.189	0.189	0	learning	0.189	0.378
0.235	0	is	0.235	0.235	0	is	0.235	0.471
0.013	0	fun	0.013	0.013	0	fun	0.013	0.027
0.006	0	exciting	0.006	0.006	0	exciting	0.006	0.012

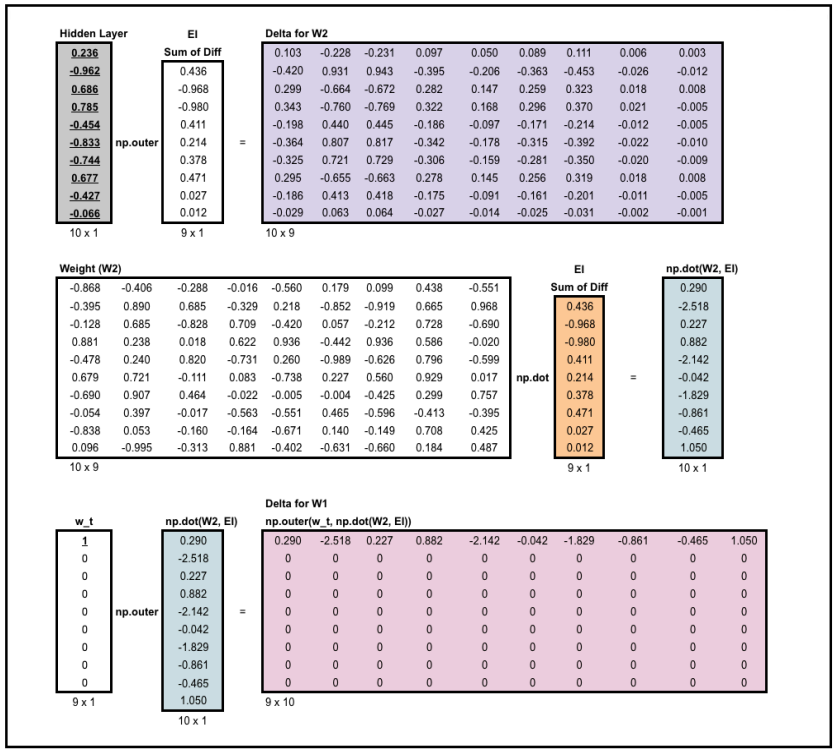
[8] Derek Chai 2018, An implementation guide to Word2Vec using Numpy and Google Sheets

An advanced embedding method: Word2Vec (9/10)

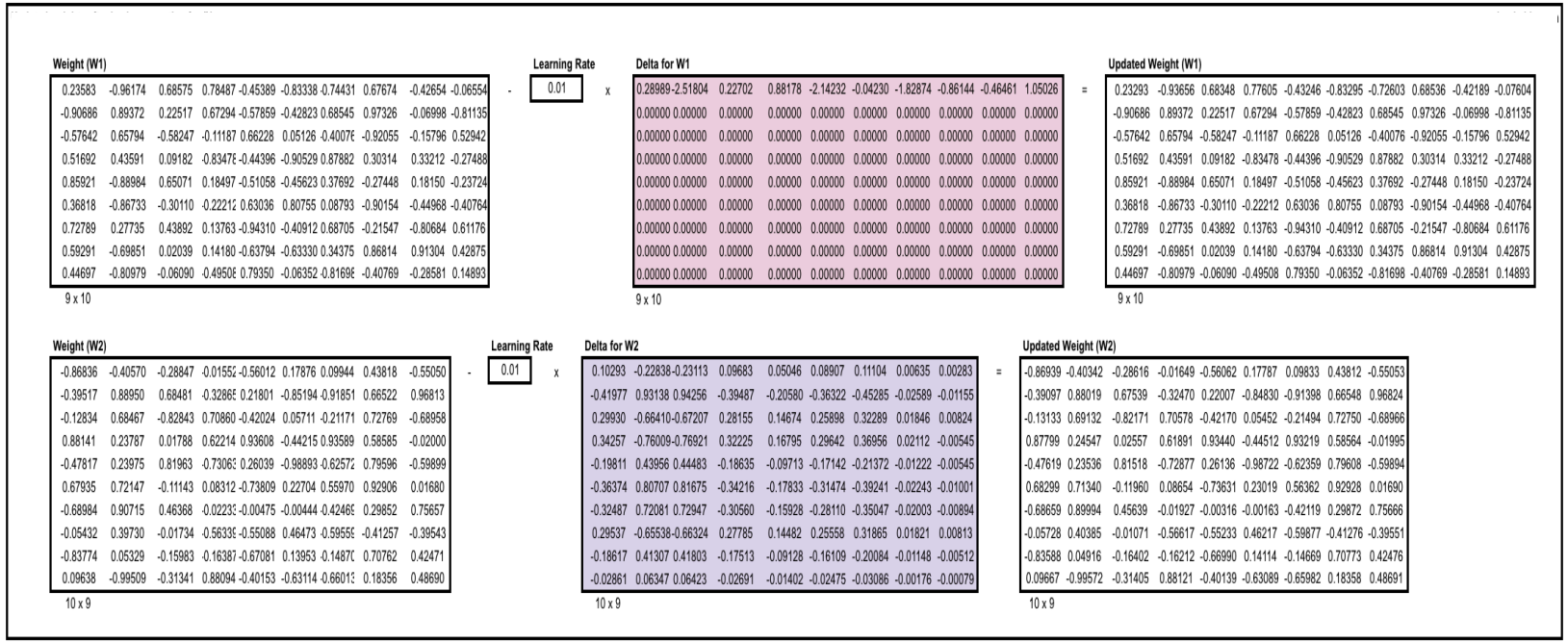
Step-by-step example[8]

natural language processing and machine learning is fun and exciting

Backpropagation



Weights update



[8] Derek Chai 2018, An implementation guide to Word2Vec using Numpy and Google Sheets

An advanced embedding method: Word2Vec (10/10)

Weaknesses of Word2Vec[8]

- Do not consider the morphology of words (subwords information) in the representation

For example, we can deduce the relationship between “dog”, “dogs”, and “dogcatcher” by their spelling. The use of another method can handle that: FastText.

- Do not separate some opposite word pairs

For example, “good” and “bad” are usually located very close to each other in the vector space, which may limit the performance of word vectors in NLP tasks like sentiment analysis.

- Takes a lot of computation time on huge text corpora

[8] Bojanowski, Grave, Joulin, Mikolov 2017 Enriching word vectors with subword information

Agenda

1. Business case
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. **Word embedding approaches**
   A. Latent Semantic Indexing (LSI) technique
   B. An advanced embedding method: Word2Vec
   C. Opening on other techniques: FastText
6. Hands-on session
7. Summary of the session

Opening on other techniques: FastText

Limits of basic skipgram model

- Hard to get good representation of non-common words
- New words are not handled
- It ignores the internal structures of words

Exemple : in french or spannish, same words (verbs) may have around forty different inflected forms

FastText principle

Improve word representation by using character level (n-gram) information.
Example : where : <“wh”, “whe”, “her”, “ere”, “re”>
Incorporate information about structure by representing words as a bag of character n-grams.

Long n-grams (n=6) are good to capture semantic information whereas shorter n-grams (n=3) are good to capture syntactic information

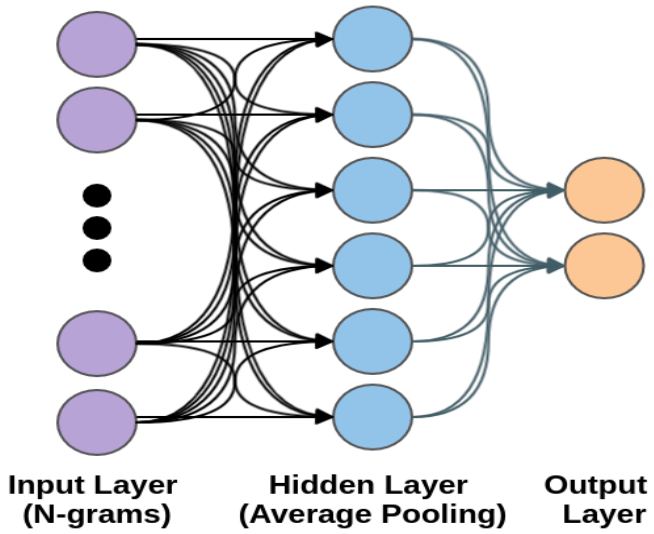
Opening on other techniques: FastText

FastText is an extension to the skipgram model in which words are represented as sum of characters n-grams[9]

Preprocessing

- A word is a bag of character. We **split a word in sequence of characters of n-grams** (bigrams/trigrams)
  Example : where : <"wh", "whe", "her", "ere", "re">
- G : set of all n-grams; G_w is the set of n-grams appearing in the word w
- Get the vector representation z_g of each n-gram as in the skipgram model before the softmax activate

Model architecture



Scoring function

$$s(w,c) = \sum_{g \in Gw} z_g{}^T v_c$$

w is the target word

V_c is the one-hot encoding vector of the context word

[9] Bojanowski, Grave, Joulin, Mikolov 2017 Enriching word vectors with subword information

Agenda

1. Business case
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. **Hands-on session**
7. Summary of the session

Hands-on 2

30’

Let’s see Word2Vec applied to your data during this practical session.

Agenda

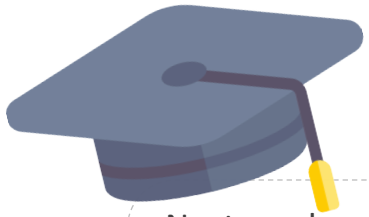
1. Business case
2. Governance bodies
3. Back to TF-IDF approach
4. Word embedding definition
5. Word embedding approaches
    A. Latent Semantic Indexing (LSI) technique
    B. An advanced embedding method: Word2Vec
    C. Opening on other techniques: FastText
6. Hands-on session
7. **Summary of the session**

Summary of the session – To remember

- An embedding is a representation in which a word is associated to a vector of numeric
- We can perform calculations and operations on embedding matrices
- LSI is a technique of dimension reduction performed with SVD for text meaning analysis
- Word2Vec is an embedding method which embark a neural networks with context and target word notions
- Cbow (context ➔ target) and Skipgram (target ➔ context) are various ways of applying Word2Vec
- FastText is an advanced form of Skigram model in which characters n-grams of a word are considered rather than the word itself. A word embedding vector is the sum/average of its inner characters embedding vectors

Work for next sessions

to be shared by Sunday February 21st, 2021



Business

Next week you'll make a restitution about what you've seen until now.

- You are working in a consulting firm
- We are your clients
- According to what is mentionned in **Slide 8**

Thank you to send us your presentation
by Sunday 21st evening
to *sim.bozko@capgemini.com and svetlana.ollivier@capgemini.com*

As a reminder, this presentation we'll be evaluated and be 20% of your total final mark.

Data-Science

To practice what we learnt today, for next session, you'll have to :

- Apply LSI on your scraped data
- Build a simple classification model using the embedding matrix and the rating of the review as target variable with a modern ML technique
- Carry out the same analysis using FastText with the gensim package (optional)
- Compare the performances of your models with two or three type of embeddings (TF-IDF, LSI, Word2Vec…)

We expect you to send your notebook file by **Sunday 21st evening** to *marie.vachelard@capgemini.com* and *augustin.mony@capgemini.com*

If you have any questions, feel free to contact us through the slack channel or by email.

References

[1] Goldberg, Yoav. Neural network methods for natural language processing. Synthesis Lectures on Human Language Technologies, 2017, vol. 10, no 1, p. 1-309.

[2] Thomas Landauer and Susan Dumais (2008) Latent semantic analysis. Scholarpedia, 3(11):4356

[3] Mikolov, Yih, Zweig, 2013, Linguistic Regularities in Continuous Space Word Representations

[4] Dhruvil Karani 2018, Introduction to word embedding and word2vec

[5] Ronxin Demo

[6] Mikolov, Chen, Corrado, Dean 2013a, Efficient estimation of Words Representationsin Vector Space

[7] Xin Rong 2016, Word2Vec Parameter Learning Explained

[8] Derek Chai 2018, An implementation guide to Word2Vec using Numpy and Google Sheets

[9] Bojanowski, Grave, Joulin, Mikolov 2017, Enriching word vectors with subword information

Course evaluation

Did you like that first course ? It’s time to share your feedbacks !

Thank you for your attention

See you next week

GOODBYE !

February 8th, 2021