Unit 3.

Language Model

Language Model

Language model refers to model that predict or generate the next component by assigning probability to elements of language (letter, word, morpheme, string (sentence), paragraph etc.).

- Language model is divided into statistical language model (SLM) and deep learning language model based on artificial neural network. Essentially language models, based on a given word, predicts the next word or combination of words, and such function can solve numerous natural language processing problems such as document generation, machine translation, document summarization etc.

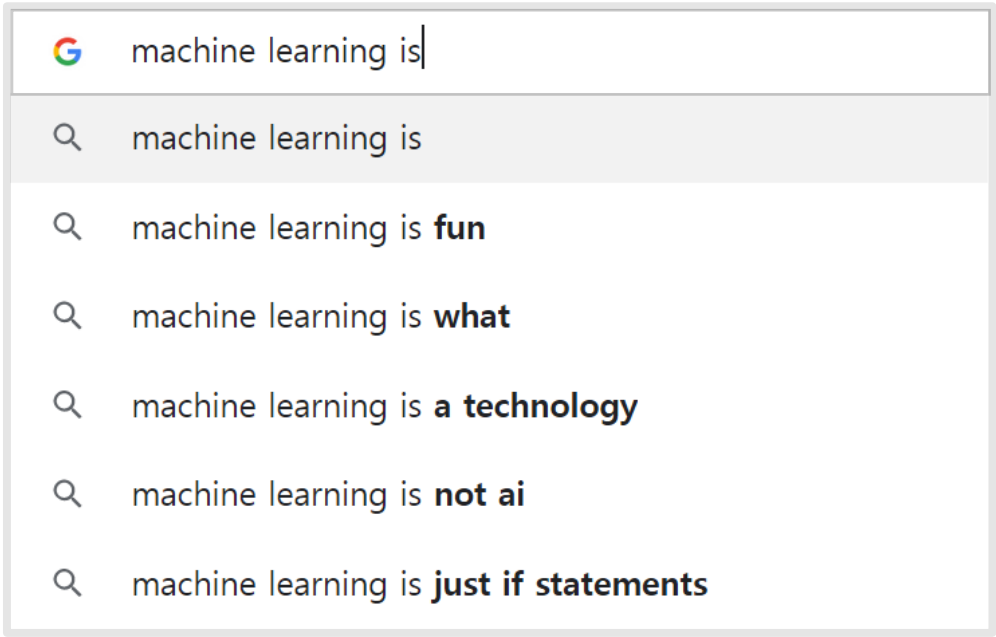
3.1. Language Model

About language model:

- Predicts the probability of a sequence: $P(w\_1,w\_2,w\_3,\ldots,w\_i)$
  Caution: The sub-index of w means the actual time order that cannot be changed.
- Given a sequence of words $\{w\_1,w\_2,w\_3,\ldots,w\_(i-1)\}$ what is the probability of w_i?
  $P(w\_i \mid w\_1,w\_2,w\_3,\ldots,w\_(i-1))$?
- Data sparsity is a major problem, because most (long) sequences appear very infrequently.
- Practical applications: machine translation, speech recognition, spell correction, autofill, etc.

About language model:

Ex In the search engine:



Probability of a long sequence

- A joint probability can be expanded as following:

$$P(w_1, w_2, w_3, \ldots, w_m) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)P(w_4|w_1, w_2, w_3) \cdots P(w_m|w_1, w_2, \ldots, w_{m-1})$$

$$= \prod_{i=1}^{m} P(w_i|w_1, \ldots, w_{i-1})$$

Ex P (three little pigs lived happily)?

$\Rightarrow$

w_1	w_2	w_3	w_4	w_5
“three”	“little”	“pigs”	“lived”	“happily”

$$P(three\ little\ pigs\ lived\ happily)$$

$$= P(three)P(little|three)P(pigs|three\ little)P(lived|three\ little\ pigs)P(happily|three\ little\ pigs\ lived)$$

n-Grams:

- Given a text sequence, n-Grams can be constructed by sliding a “moving window” of length = n.

Ex “three little pigs lived happily”

→ $n = 1$, Unigrams = [“three”, “little”, “pigs’, “lived”, “happily”]

→ $n = 2$, Bigrams = [“three little”, “little pigs”, “pigs lived”, “lived happily”]

→ $n = 3$, Trigrams = [“three little pigs”, “little pigs lived”, “pigs lived happily”]

“three little pigs lived happily”

“three little pigs lived happily”

“three little pigs lived happily”

n-Gram approximations:

- As the sequence grows, the probabilities become harder to estimate due to the data sparsity:

$$P(w_i|w_1, w_2, w_3, \dots, w_{i-1}) = \frac{Count(w_1, w_2, w_3, \dots, w_i)}{Count(w_1, w_2, w_3, \dots, w_{i-1})}$$

- Instead of an exact estimation of probabilities, we can do the so-called n -Gram approximation:

$$P(w_1, w_2, w_3, \dots, w_m) \approx \prod_{i=1}^{m} P(w_i|w_{i-(n-1)}, \dots, w_{i-1})$$

which can be compared with the following exact relation.

$$P(w_1, w_2, w_3, \dots, w_m) = \prod_{i=1}^{m} P(w_i|w_1, \dots, w_{i-1})$$

- => Usually the n above is a small positive number ≅1, 2, 3,…

n-Gram approximations:

- When n=1, it is the Unigram approximation:

$$P(w_1, w_2, w_3, \ldots, w_m) \approx P(w_1)P(w_2)P(w_3)\cdots P(w_m)$$

- When n=2, it is the Bigram approximation:

$$P(w_1, w_2, w_3, \ldots, w_m) \approx P(w_1)P(w_2|w_1)P(w_3|w_2)\cdots P(w_m|w_{m-1})$$

- When n=3, it is the Trigram approximation:

$$P(w_1, w_2, w_3, \ldots, w_m) \approx P(w_1)P(w_2|w_1)P(w_3|w_2, w_1)P(w_4|w_3, w_2)\cdots P(w_m|w_{m-1}, w_{m-2})$$

Ex Bigram approximation for **Sequence** = *"three little pigs lived happily"*

$$P(\boldsymbol{Sequence}) \approx P(three)P(little|three)P(pigs|little)P(lived|pigs)P(hapilly|lived)$$

Coding Exercise #0509

Follow practice steps on 'ex_0509.ipynb' file

Unit 3.

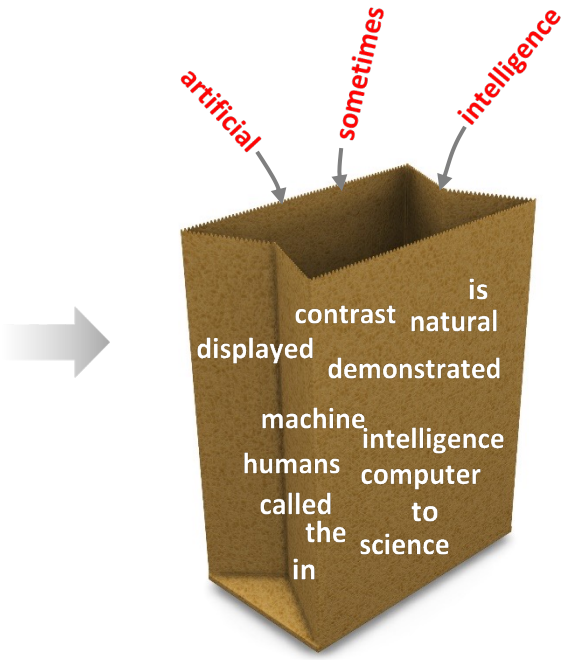
Language Model

Representation Model

Bag-of-Words (BOW) model:

- A document is represented by a collections of its words.
- Word ordering and grammar are ignored.
- Only the word frequencies matter.

Ex "In computer science, artificial intelligence, sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans."



Bag-of-Words (BOW) model:

Ex "It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness."

⇒ after removing the stop words such as "it", "the", and "of", the BOW can be expressed as an array.

age	best	foolishness	times	was	wisdom	worst
2	1	1	2	1	1	1

Frecuencia de la palabra en el documento

3.2. Representation Model

Document-Term Matrix (DTM) and Term-Document Matrix (TDM):

- Documents expressed as BOW are the rows of DTM.
- Documents expressed as BOW are the columns of TDM.

	Feature #1	Feature #2	Feature #3	Feature #4	...
Document #1	1	0	1	0	0
Document #2	0	0	2	0	0
Document #3	0	1	0	0	1
Document #4	0	0	0	1	0
⋮	0	0	1	0	0

DTM

	Document #1	Document #2	Document #3	Document #4	...
Feature #1	1	0	0	0	0
Feature #2	0	0	1	0	0
Feature #3	1	2	0	0	1
Feature #4	0	0	0	1	0
⋮	0	0	1	0	0

TDM

Document-Term Matrix (DTM) and Term-Document Matrix (TDM):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics"

Document #2: "machine classification learning performance"

Document #3: "machine classification, machine learning, machine performance"

DTM =

	learning	intelligence	machine	statistics	classification	performance
Doc. #1	2	1	1	1	0	0
Doc. #2	1	0	1	0	1	1
Doc. #3	1	0	3	0	1	1

Document-Term Matrix (DTM) and Term-Document Matrix (TDM):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics"

Document #2: "machine classification learning performance"

Document #3: "machine classification, machine learning, machine performance"

TDM =

	Doc. #1	Doc. #2	Doc. #3
learning	2	1	1
intelligence	1	0	0
machine	1	1	3
statistics	1	0	0
classification	0	1	1
performance	0	1	1

Term Frequency (TF):

- Indicates the relative importance of each word (term) within a document.
- A frequently occurring word within a short document would have a large TF value.

$$TF(word, document) = \frac{Frequency\ of\ the\ word\ within\ the\ document}{The\ document\ length}$$

- TF has to be calculated per word and per document.

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

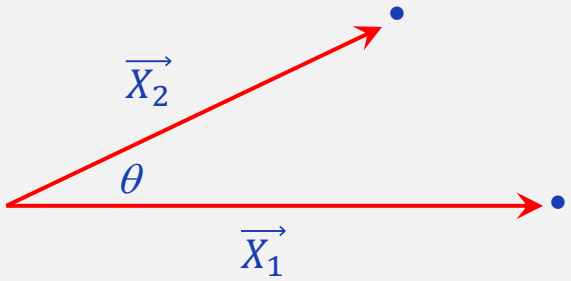
Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3
learning	2/5 = 0.4	1/4 = 0.25	1/6 = 0.17
intelligence	1/5 = 0.2	0	0
machine	1/5 = 0.2	1/4 = 0.25	3/6 = 0.5
statistics	1/5 = 0.2	0	0
classification	0	1/4 = 0.25	1/6 = 0.17
performance	0	1/4 = 0.25	1/6 = 0.17

Cosine similarity:

- Documents are vectors. The similarity between two documents can be quantified.



$$\text{Cosine similarity is } Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$$

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0.4	0.25	0.17	0
intelligence	0.2	0	0	1/2
machine	0.2	0.25	0.5	1/2
statistics	0.2	0	0	0
classification	0	0.25	0.17	0
performance	0	0.25	0.17	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine intelligence"?

Q1: (0, 0.5, 0.5, 0, 0, 0, 0) |Q1| =0.71
Doc 1: (0.4, 0.2, 0.2, 0.2, 0, 0) |Doc1|=0.53
Doc 2: (0.25, 0, 0.25, 0, 0.25, 0.25) |Doc2|=0.5
Doc 3: (0.17, 0, 0.5 ,0 ,0.17 ,0.17) |Doc3|=0.58

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)=0.2/(0.71*0.53)= 0.53
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)=0.12/(0.71*0.5)= 0.35
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)=0.25/(0.71*0.58)= 0.61

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0.4	0.25	0.17	1/2
intelligence	0.2	0	0	0
machine	0.2	0.25	0.5	1/2
statistics	0.2	0	0	0
classification	0	0.25	0.17	0
performance	0	0.25	0.17	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine learning"?

Q1: (0.5, 0, 0.5, 0, 0, 0, 0) |Q1| =0.71
Doc 1: (0.4, 0.2, 0.2, 0.2, 0, 0) |Doc1|=0.53
Doc 2: (0.25, 0, 0.25, 0, 0.25, 0.25) |Doc2|=0.5
Doc 3: (0.17, 0, 0.5 ,0 ,0.17 ,0.17) |Doc3|=0.58

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)= 0.3/(0.71*0.53)= 0.80
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)= 0.25/(0.71*0.5)= 0.71
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)= 0.34/(0.71*0.58)= 0.82

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0.4	0.25	0.17	1/3
intelligence	0.2	0	0	0
machine	0.2	0.25	0.5	1/3
statistics	0.2	0	0	0
classification	0	0.25	0.17	1/3
performance	0	0.25	0.17	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine learning classification"?

Q1: (0.33, 0, 0.33, 0, 0, 0.33, 0) |Q1| =0.57
Doc 1: (0.4, 0.2, 0.2, 0.2, 0, 0) |Doc1|=0.53
Doc 2: (0.25, 0, 0.25, 0, 0.25, 0.25) |Doc2|=0.5
Doc 3: (0.17, 0, 0.5 ,0 ,0.17 ,0.17) |Doc3|=0.58

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)= 0.2/(0.57*0.53)= 0.65
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)= 0.25/(0.57*0.5)= 0.87
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)= 0.28/(0.57*0.58)= 0.84

Document Frequency (DF) and Inverse Document Frequency (IDF):

- DF: the number of documents where a particular word appears.
- N: total number of documents
- IDF: a measure of rarity and information carried by a particular word.

$$IDF(word) = Log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ that\ include\ the\ word}\right)$$

- IDF is a property of the corpus and has to be calculate per word only.

- $TD\text{-}IDF(t,d) = TF_{t,d} \times IDF_t = TF_{t,d} \times \log(N/df_t)$

The formula can be adjusted to avoid a division by zero, in case df_t was 0: $TF_{t,d} \times \log(N+1/(1 + df_t))$

Document Frequency (DF) and Inverse Document Frequency (IDF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics"

Document #1: "machine classification learning performance"

Document #1: "machine classification machine learning machine performance"

IDF =

	DF	IDF
learning	3	Log(3/3) = 0
intelligence	1	Log(3/1) = 0.48
machine	3	Log(3/3) = 0
statistics	1	Log(3/1) = 0.48
classification	2	Log(3/2) = 0.18
performance	2	Log(3/2) = 0.18

TF IDF representation:

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics"

Document #1: "machine classification learning performance"

Document #1: "machine classification machine learning machine performance"

TF IDF =

	Doc. #1	Doc. #2	Doc. #3
learning	0.4	0.25	0.17
intelligence	0.2	0	0
machine	0.2	0.25	0.5
statistics	0.2	0	0
classification	0	0.25	0.17
performance	0	0.25	0.17

×

	IDF
learning	0
intelligence	0.48
machine	0
statistics	0.48
classification	0.18
performance	0.18

=

	Doc. #1	Doc. #2	Doc. #3
learning	0	0	0
intelligence	0.095	0	0
machine	0	0	0
statistics	0.095	0	0
classification	0	0.044	0.03
performance	0	0.044	0.03

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0	0	0	0
intelligence	0.095	0	0	1/2
machine	0	0	0	1/2
statistics	0.095	0	0	0
classification	0	0.044	0.03	0
performance	0	0.044	0.03	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine intelligence"?

Q1: (0, 0.5, 0.5, 0, 0, 0, 0) |Q1| =0.71
Doc 1: (0, 0.095, 0, 0.095, 0, 0) |Doc1|=0.13
Doc 2: (0, 0, 0, 0, 0.044, 0.044) |Doc2|=0.06
Doc 3: (0, 0, 0 ,0 ,0.03 ,0.03) |Doc3|=0.04

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)=0.05/(0.71*0.13)= 0.5
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)=0.00/(0.71*0.06)= 0.0
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)=0.00/(0.71*0.04)= 0.0

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0	0	0	1/2
intelligence	0.095	0	0	0
machine	0	0	0	1/2
statistics	0.095	0	0	0
classification	0	0.044	0.03	0
performance	0	0.044	0.03	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine learning"?

Q1: (0.5, 0, 0.5, 0, 0, 0, 0) |Q1| =0.71
Doc 1: (0, 0.095, 0, 0.095, 0, 0) |Doc1|=0.13
Doc 2: (0, 0, 0, 0, 0.044, 0.044) |Doc2|=0.06
Doc 3: (0, 0, 0 ,0 ,0.03 ,0.03) |Doc3|=0.04

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)=0.00/(0.71*0.13)= 0.0
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)=0.00/(0.71*0.06)= 0.0
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)=0.00/(0.71*0.04)= 0.0

Similarity.ipynb

Team Frequency (TF):

Ex Let's suppose the following pre-processed documents.

Document #1: "learning intelligence machine learning statistics" ⇒ length = 5

Document #2: "machine classification learning performance" ⇒ length = 4

Document #3: "machine classification machine learning machine performance" ⇒ length = 6

TF =

	Doc. #1	Doc. #2	Doc. #3	Q1
learning	0	0	0	1/3
intelligence	0.095	0	0	0
machine	0	0	0	1/3
statistics	0.095	0	0	0
classification	0	0.044	0.03	1/3
performance	0	0.044	0.03	0

Cosine similarity is $Cos(\theta) = \frac{\overrightarrow{X_1} \cdot \overrightarrow{X_2}}{|\overrightarrow{X_1}||\overrightarrow{X_2}|}$

Which document is more relevant for "machine learning classification"?

Q1: (0.33, 0, 0.33, 0, 0, 0.33, 0) |Q1| =0.57
Doc 1: (0, 0.095, 0, 0.095, 0, 0) |Doc1|=0.13
Doc 2: (0, 0, 0, 0, 0.044, 0.044) |Doc2|=0.06
Doc 3: (0, 0, 0 ,0 ,0.03 ,0.03) |Doc3|=0.04

Sim(Q1,Doc1) = (Q1*Doc1)/(|Q1|*|Doc1|)=0.00/(0.57*0.13)= 0.0
Sim(Q1,Doc2) = (Q1*Doc2)/(|Q1|*|Doc2|)=0.01/(0.57*0.06)= 0.41
Sim(Q1,Doc3) = (Q1*Doc3)/(|Q1|*|Doc3|)=0.01/(0.57*0.04)= 0.41

Coding Exercise #0510

Follow practice steps on 'ex_0510.ipynb' file

Unit 3.

Language Model

Naïve Bayes Classifier

Naïve Bayes classifier using the BOW model:

- For convenience, let's suppose that there are two document types A and B. The document types can be, for example, A= "spam" and B = "no spam".
- Let's apply the BOW model: the bags A and B contain the tokenized words.
- Applying the Bayes' theorem, we have:

$$P(A|w_1, w_2, w_3, \dots) = \frac{P(w_1, w_2, w_3, \dots|A)P(A)}{P(w_1, w_2, w_3, \dots)}$$

$$P(B|w_1, w_2, w_3, \dots) = \frac{P(w_1, w_2, w_3, \dots|B)P(B)}{P(w_1, w_2, w_3, \dots)}$$

Caution: The sub-index of w serves only a labeling purpose. Here, the words are not ordered.

Naïve Bayes classifier using the BOW model:

- Prediction based on the comparison between $P(A|w\_1,w\_2,w\_3,\ldots)$ and $P(B|w\_1,w\_2,w\_3,\ldots)$.
- For the comparison, only the relative difference matters. Question: Which probability is higher?
- For the comparison, we do not need the common denominator $P(w\_1,w\_2,w\_3,\ldots)$.

$$P(A|w_1, w_2, w_3, \ldots) \sim P(w_1, w_2, w_3, \ldots|A)P(A)$$
$$P(B|w_1, w_2, w_3, \ldots) \sim P(w_1, w_2, w_3, \ldots|B)P(B)$$

- In the BOW model, the words occur independently from each other.
  Thus, we can expand in the following way.

$$P(A|w_1, w_2, w_3, \ldots) \sim P(w_1|A)P(w_2|A)P(w_3|A)\cdots P(A)$$
$$P(B|w_1, w_2, w_3, \ldots) \sim P(w_1|B)P(w_2|B)P(w_3|B)\cdots P(B)$$

Naïve Bayes classifier using the BOW model:

- Instead of comparing the probabilities, we can compare the logarithms of probabilities.
- Applying the $Log()$ on both sides of the equal sign, we have:

$$Log(P(A|w_1, w_2, w_3, \dots)) \sim Log\big(P(w_1|A)\big) + Log\big(P(w_2|A)\big) + Log\big(P(w_3|A)\big) + \cdots + Log(P(A))$$

$$Log(P(B|w_1, w_2, w_3, \dots)) \sim Log\big(P(w_1|B)\big) + Log\big(P(w_2|B)\big) + Log\big(P(w_3|B)\big) + \cdots + Log(P(B))$$

- If we balance the training set such that the number of type A = number of type B, then $Log(P(A)) = Log(P(B))$. So, we can also drop these terms in the comparison.

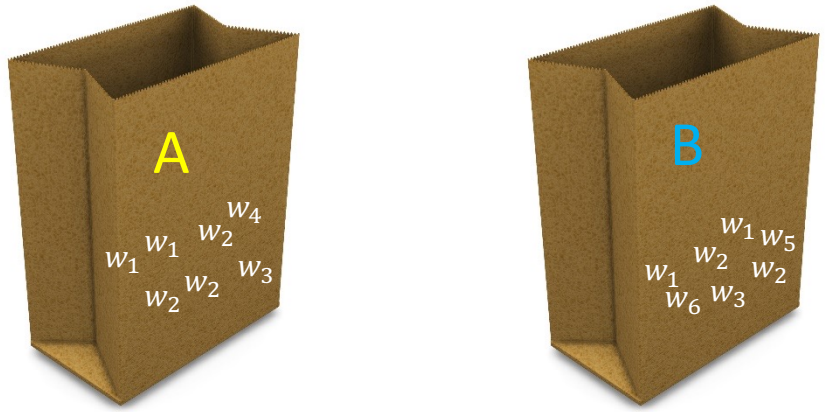
Naïve Bayes classifier using the BOW model:

- Training step:

  1) For each word in the bag A, calculate the probabilities $P(w\_i|A)$ and their logarithms $Log(P(w\_i|A))$.

  2) For each word in the bag B, calculate the probabilities $P(w\_i|B)$ and their logarithms $Log(P(w\_i|B))$.

  3) Save for later use the logarithmic probabilities calculated in the steps 1) and 2).



Naïve Bayes classifier using the BOW model:

- Prediction step:

1) Given a test document made up of words $w'\_1, w'\_2, w'\_3, \ldots$ add their logarithmic probabilities:

$$LogProbA = Log\left(P(w'_1|A)\right) + Log\left(P(w'_2|A)\right) + Log\left(P(w'_3|A)\right) + \cdots$$

$$LogProbB = Log\left(P(w'_1|B)\right) + Log\left(P(w'_2|B)\right) + Log\left(P(w'_3|B)\right) + \cdots$$

2) If $LogProbA > LogProbB$: then the test document is predicted as type A.
If $LogProbA < LogProbB$: then the test document is predicted as type B.

Classification Analysis

Classification analysis using the TF IDF model:

- In the TF IDF model:

  → document ≅ observation.

  → word (W_i) ≅ explanatory variable (X_i).

- If the data is labeled (response Y), we can do predictive analysis with the classification algorithms such as logistic regression, KNN, decision tree, Random Forest, etc.

	X_1	X_2	X_3	$\cdots$	Y
Obs. #1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Obs. #2	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Obs. #3	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\vdots$					

	W_1	W_2	W_3	$\cdots$	Y
Doc. #1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Doc. #2	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Doc. #3	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\vdots$					

Coding Exercise #0511

Follow practice steps on 'ex_0511.ipynb' file

Unit 3.

Language Model

Word Meaning

Meaning of "mouse" in Wordnet

- mouse (N) ← Lemma
  1. Any of the numerous small rodents …
  2. A hand-operated electronic device that controls a cursor …
  3. …

Senses

Polysemy

Some relationships between senses

- **Synonymy** (the same meaning)
  1. couch / sofa
  2. car / automobile
  3. big / large
  4. water / H2O
- **Similarity** (similar meanings or uses)
  1. car / bicycle
  2. cat / dog
  3. coffee / tea
- **Antonymy** (the opposite meaning)
  1. dark / light
  2. up / down
  3. hot / cold
- **Word relatedness** (different meanings used in a semantic domain or field)
  1. coffee / cup
  2. surgeon / nurse / hospital
  3. menu / food / restaurant

Word semantics and embeddings. Dan Jurafsky. https://web.stanford.edu/~jurafsky/

Word Meaning

Vector semantics

- Meaning of a word depends on its relationship with other words (or meanings)
- Meaning is a point in a multidimensional space based on distribution
- **Each word = a vector**
- Similar words are **nearby in semantic space**
- Semantic space is built automatically **by seeing wich words are nearby in text**

Word Meaning

Two words are similar in meaning if their context vectors are similar

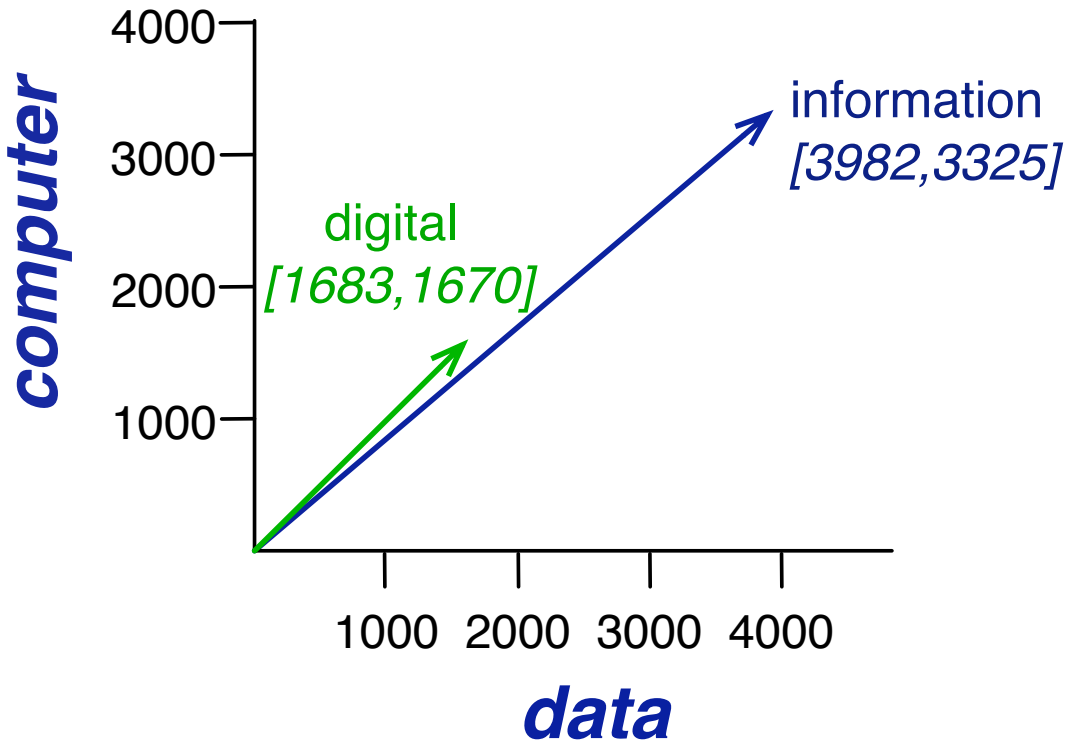
is traditionally followed by	**cherry**	pie, a traditional dessert
often mixed, such as	**strawberry**	rhubarb pie. Apple pie
computer peripherals and personal	**digital**	assistants. These devices usually
a computer. This includes	**information**	available on the internet

word-word matrix

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

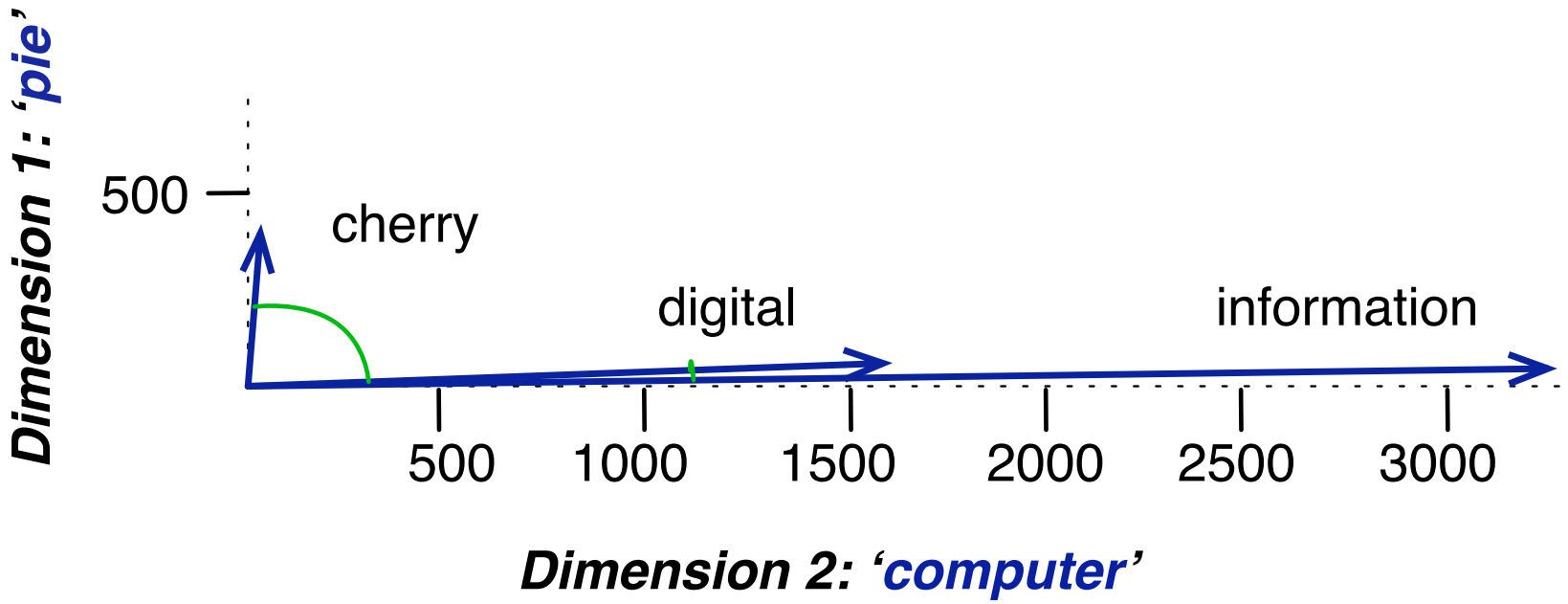
Word Meaning

- Example of projection in a bidimensional space



Word Meaning

- The most used similarity metric is **cosine**



One-hot-encoding vs word embedding:

- We notice obvious problems with the one-hot-encoding representation.
- To improve, the “word embedding” is introduced which is a distributed representation method.

One-Hot-Encoding	Embedding
The dimension of the vector space is large. The dimension is as large as the vocabulary size. Dimension = |V| = 20,000 to 50,000	The dimension of the vector space is limited. Dimension = 50 - 1000
Vectors are sparse; they are mostly filled with 0s that carry no information.	Vectors are dense. Every vector element carries some information.
No semantic relationship among the vectors. The vectors are orthogonal to each other.	Semantic relationship among the vectors.

- There are also “paragraph embedding” and “document embedding” representations.
- We will call “dense vector” or “embedding vector” interchangeably.

https://projector.tensorflow.org/ https://jalammar.github.io/illustrated-word2vec/

One-hot-encoding vs word embedding:

Ex Given a sentence “I eat an apple every morning”, let’s suppose that the words are indexed as:

I : 3
Eat : 0
An : 2
Apple : 1
Every : 4
Morning : 5

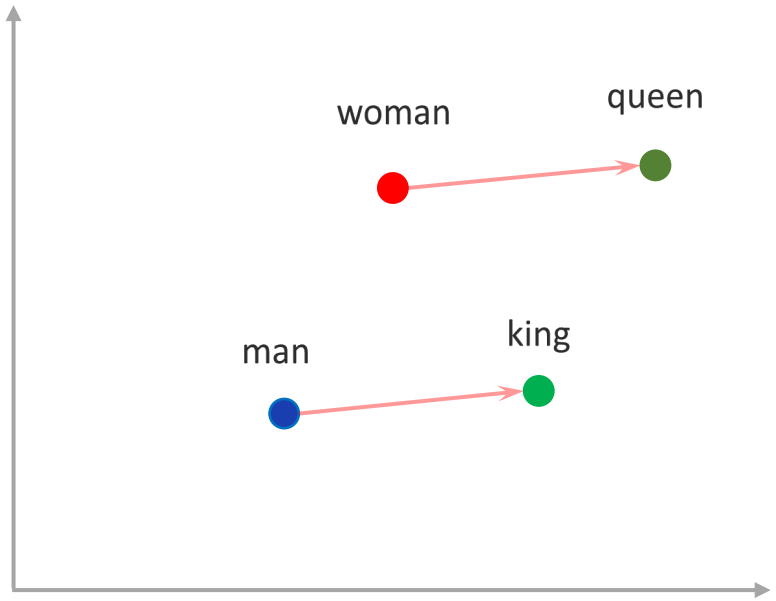
The words would have the following one-hot-encoding representations:

I : [0 0 0 1 0 0]
Eat : [1 0 0 0 0 0]
An : [0 0 1 0 0 0]
Apple : [0 1 0 0 0 0]
Every : [0 0 0 0 1 0]
Morning : [0 0 0 0 0 1]

Word embedding (Word2Vec):

Ejemplo_Embeddings.ipynb

- Among the dense vectors, relationships such as following are established:



$$queen – woman = king – man$$

man is a king as woman is a queen

woman + king – man = queen

Coding Exercise #0511

Follow practice steps on 'Ejercicio_Word2vec.ipynb' file