Web Content Mining
Generalities, Preprocessing and Word Associations
— Session 1 —
Text Mining and Analytics

- Text mining ~ Text Analytics
- Turn text data into high-quality information or actionable knowledge.
  - Minimizes human effort (on consuming text data)
  - Supplied knowledge for optimal decision making
- Related to text retrieval, which is an essential component in any text mining system
  - Text retrieval can be a preprocessor for text mining
  - Text retrieval is needed for knowledge provenance
Why Text Mining is hard?

- I ate pizza with friends = I ate pizza with olives?
  - Only one different word
  - Friends = Olives?
- I ate pizza with friends = friends and I shared some pizza?
  - 3 similar words
  - Different verb
  - 3/5 vs 3/6 words
  - Similar semantic sense
Text vs. Non-Text data

Real world → Sensor → Data

Weather → Thermometer → 3C, 15F, etc
Locations → Geo Sensor → 41N and 120W
Networks → Network Sensor → 0100010001100

Real World → Human Sensor → Text
The General Problem of Data Mining

- Actionnable knowledge

Sensor 1
Sensor 2
... Sensor k

Non text data
- Numerical
- Categorical
- Relational
- Images
- Videos

Text

Data Mining

- Graph Mining

Text Mining
The General Problem of Data Mining

Actionnable knowledge

Sensor 1
Sensor 2
... 
Sensor k

Non text data
Numerical
Categorical
Relational
Images
Videos

TEXT

Data Mining

Graph Mining

Text Mining
Landscape of Text Mining and Analytics

1. Mining knowledge about language
2. Mining content of text data
3. Mining knowledge about the observer
4. Infer other real-world variables (predictive analytics)

Real World → Observed World

Perceive

Express (english)

Text Data

1. Mining knowledge about language
2. Mining content of text data
3. Mining knowledge about the observer
4. Infer other real-world variables (predictive analytics)
Landscape of Text Mining and Analytics

Real World → Perceive → Observed World → Express (english) → Text Data

- Text based prediction
- Topic mining and analysis
- Opinion mining & Sentiment analysis
- Word association mining & analysis

NLP & text representation
Basics of Text Mining

- **Collections**
  - Documents are compressed
  - Uncommon formats
  - Sometimes they just don’t exist

- **Documents**
  - A lot of preprocessing (encoding, cleaning, splitting, etc.)
  - Granularity level (whole document, paragraph, sentence, etc.)

- **Words**
  - Steaming, upper/lower case, frequent (stopwords) and infrequent words, special characters, dates, prices, names, emails, etc.

- **General**
  - Language: main tasks have been addressed for English (with no 100% performance), but many other languages are far from English and many of them are completely without resources (regional languages)
  - Lot of formulas and concepts
  - Task is the main factor
  - Tools: python (NLKT), java (OpenNLP), c, etc.
Landscape of Text Mining and Analytics

Real World → Perceive → Observed World → Express (english) → Text Data

Text based prediction

Topic mining and analysis

Opinion mining & Sentiment analysis

Word association mining & analysis

NLP & text representation
Basic Concepts in NLP

Lexical analysis (Part-of-speech tagging)

A dog is chasing a boy on the playground

Det Noun Aux Verb Det Noun Prep Det Noun

Noun Phrase Complex verb Noun Phrase Noun Phrase

Verb Phrase Prep Phrase Verb Phrase

Sentence

Semantic Analysis

Dog (d1) Boy (b1) Playground(p1) Chasing(d1,b1,p1) + Scared(x) if Chasing(_,x,_) => Scared(b1)
NLP is difficult!

- Natural language is designed to make human communication efficient. As a result,
  - We omit a lot of common sense knowledge, which we assume the hearer/reader possesses.
  - We keep a lot of ambiguities, which we assume the hearer/reader knows how to resolve.
- This makes EVERY step in NLP hard
  - Ambiguity is a killer!
  - Common sense reasoning is pre-required.
Examples of Challenges

- **Word-level ambiguity:**
  - “design” can be a noun or a verb (ambiguous POS)
  - “root” has multiple meanings (ambiguous sense)

- **Syntactic ambiguity:**
  - “natural language processing” (modification)
  - “A man saw a boy with a telescope” (PP attachment)
  - Anaphora resolution: “John persuaded Bill to buy a TV for himself” (himself = John or Bill?)
  - Presupposition: “He has quite smoking” implies that he smoked before.
What we can’t do

- 100 % POS tagging
  - ”He turned off the highway” vs “He turned off the fan”
- General complete parsing
  - “A man saw a boy with a telescope”
- Precise deep semantic analysis
  - Will we ever be able to precisely define meaning of “own” in “John owns a restaurant”?

Robust and general NLP tends to be shallow while deep understanding doesn’t scale up.
Take away message

- NLP is the foundation for text mining
- Computers are far from being able to understand natural language
  - Deep NLP requires common sense knowledge and inferences, thus only working for very limited domains
  - Shallow NLP based on statistical methods can be done in large scale and is thus more broadly applicable
- In practice: statistical NLP as the basis, while humans provide help as needed
A dog is chasing a boy on the playground.

String of chars.
Seq. of words
POS tags
Syntactic structures
Entities and Relations
Logic predicates

A dog
CHASE
a boy
ON
the playground

Animal
Person
Location

Dog (d1), Boy(b1), Playground (p1), Chasing (d1,b1,p1)
# Text Representation and Enabled Analysis

<table>
<thead>
<tr>
<th>Text Rep</th>
<th>Generality</th>
<th>Enabled Analysis</th>
<th>Examples of Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>******</td>
<td>String processing</td>
<td>Compression</td>
</tr>
<tr>
<td>Words</td>
<td>****</td>
<td>Word relation analysis; topic analysis; sentiment analysis</td>
<td>Thesaurus discovery; topic and opinion related applications</td>
</tr>
<tr>
<td>Syntactic structures</td>
<td>***</td>
<td>Syntactic graph analysis</td>
<td>Stylistic analysis; structure-based feature extraction</td>
</tr>
<tr>
<td>Entities &amp; relations</td>
<td>**</td>
<td>Knowledge graph analysis; information network analysis</td>
<td>Discovery of knowledge ans opinions about specific entities</td>
</tr>
<tr>
<td>Logic predicates</td>
<td>*</td>
<td>Integrative analysis of scattered knowledge; logic inference</td>
<td>Knowledge assistant for biologist</td>
</tr>
</tbody>
</table>
Take away message

- Text representations determine what kind of mining algorithms can be applied
- Multiple ways of representing text are possible
  - String, words, syntactic structures, entity-relation graphs, predicates, etc.
  - Can/should be combined in real applications
- This course focuses mainly on word-based representation
  - General and robust: applicable to any natural language
  - No/little manual effort
  - “Surprisingly” powerful for many applications (not all!)
  - Can be combined with more sophisticated representations
- Tools
  - Python NLTK
Landscape of Text Mining and Analytics

- Text based prediction
- Topic mining and analysis
- Opinion mining & Sentiment analysis
- Word association mining & analysis
- NLP & text representation
Basic word relations

- Paradigmatic: A & B have paradigmatic relation if they can be substituted for each other (i.e., A & B are in the same class)
  - e.g.; “cat” and “dog”; “Monday” and “Tuesday”

- Syntagmatic: A & B have syntagmatic relation if they can be combined with each other (i.e., A & B are related semantically)
  - e.g., “cat” and “sit”; “car” and “drive”

- These two basic and complementary relations can be generalized to describe relations of any items in a language
Why mine word associations?

- They are useful for improving accuracy of many NLP tasks
  - POS tagging, parsing, entity recognition, acronym expansion
  - Grammar learning
- They are directly useful for many applications in text retrieval and mining
  - Text retrieval (e.g., use word associations to suggest a variation of a query)
  - Automatic construction of topic map of browsing: words as nodes ans associations as edges
  - Compare and summarize opinions (e.g., what words are most strongly associated with ‘‘battery’’ in positive and negative reviews about iPhone7, respectively?)
Mining Word Associations: Intuitions

My cat eats fish on Saturday
His cat eats turkey on Tuesday
My dog eats meat on Sunday
His dog eats turkey on Tuesday

**Paradigmatic:**
How similar are the context ("cat") and context ("dog")?
How similar are the context ("cat") and context ("computer")?

**Syntagmatic:**
How helpful is the occurrence of "eats" for predicting the occurrence of "meat"?
How helpful is the occurrence of "eats" for predicting the occurrence of "text"?
Mining Word Associations: General Ideas

‣ Paradigmatic
  ‣ Represent each word by its context
  ‣ Compute context similarity
  ‣ Words with high context similarity likely have paradigmatic relation

‣ Syntagmatic
  ‣ Count how many times two words occur together in a context (e.g., sentence or paragraph)
  ‣ Compare their co-occurrences with their individual occurrences
  ‣ Words with high co-occurrences but relatively low individual occurrences likely have syntagmatic relation
Distributional semantics

- Comparing two words:
  - Look at all context words for word1
  - Look at all context words for word2
  - How similar are those two context collections in their entirety?

- Compare distributional representations of two words
How can we compare two context collections in their entirety?

Count how often “apple” occurs close to other words in a large text collection (corpus):

<table>
<thead>
<tr>
<th>eat</th>
<th>fall</th>
<th>ripe</th>
<th>slice</th>
<th>peel</th>
<th>tree</th>
<th>throw</th>
<th>fruit</th>
<th>pie</th>
<th>bite</th>
<th>crab</th>
</tr>
</thead>
<tbody>
<tr>
<td>794</td>
<td>244</td>
<td>47</td>
<td>221</td>
<td>208</td>
<td>160</td>
<td>145</td>
<td>156</td>
<td>109</td>
<td>104</td>
<td>88</td>
</tr>
</tbody>
</table>

Interpret counts as coordinates:

Every context word becomes a dimension.
How can we compare two context collections in their entirety?

Count how often “apple” occurs close to other words in a large text collection (corpus):

<table>
<thead>
<tr>
<th>eat</th>
<th>fall</th>
<th>ripe</th>
<th>slice</th>
<th>peel</th>
<th>tree</th>
<th>throw</th>
<th>fruit</th>
<th>pie</th>
<th>bite</th>
<th>crab</th>
</tr>
</thead>
<tbody>
<tr>
<td>794</td>
<td>244</td>
<td>47</td>
<td>221</td>
<td>208</td>
<td>160</td>
<td>145</td>
<td>156</td>
<td>109</td>
<td>104</td>
<td>88</td>
</tr>
</tbody>
</table>

Do the same for “orange”:

<table>
<thead>
<tr>
<th>eat</th>
<th>fall</th>
<th>ripe</th>
<th>slice</th>
<th>peel</th>
<th>tree</th>
<th>throw</th>
<th>fruit</th>
<th>pie</th>
<th>bite</th>
<th>crab</th>
</tr>
</thead>
<tbody>
<tr>
<td>265</td>
<td>22</td>
<td>25</td>
<td>62</td>
<td>220</td>
<td>64</td>
<td>74</td>
<td>111</td>
<td>4</td>
<td>4</td>
<td>8</td>
</tr>
</tbody>
</table>
How can we compare two context collections in their entirety?

Then visualize both count tables as vectors in the same space:

<table>
<thead>
<tr>
<th>eat</th>
<th>fall</th>
<th>ripe</th>
<th>slice</th>
<th>peel</th>
<th>tree</th>
<th>throw</th>
<th>fruit</th>
<th>pie</th>
<th>bite</th>
<th>crab</th>
</tr>
</thead>
<tbody>
<tr>
<td>794</td>
<td>244</td>
<td>47</td>
<td>221</td>
<td>208</td>
<td>160</td>
<td>145</td>
<td>156</td>
<td>109</td>
<td>104</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Similarity between two words as proximity in space.
Where can we find texts to use for making a distributional model?

- Text in electronic form!
- Newspaper articles
- Project Gutenberg: older books available for free
- Wikipedia
- Text collections prepared for language analysis:
  - Balanced corpora
  - WaC: Scrape all web pages in a particular domain
  - ELRA, LDC hold corpus collections
    - For example, large amounts of newswire reports
  - Google n-grams, Google books
What do we mean by “similarity” of vectors?

Euclidean distance as a dissimilarity measure

\[ \text{dist}(\vec{p}, \vec{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}. \]
Problem with Euclidean distance: very sensitive to word frequency!
What do we mean by “similarity” of vectors?

Use **angle** between vectors instead of point distance to get around word frequency issues

\[
\cos(\vec{p}, \vec{q}) = \frac{\sum_{i=1}^{n} p_i \cdot q_i}{\sqrt{\sum_{i=1}^{n} p_i^2} \cdot \sqrt{\sum_{i=1}^{n} q_i^2}}
\]
Some counts for “letter” in “Pride and Prejudice”. What do you notice?

<table>
<thead>
<tr>
<th>the</th>
<th>to</th>
<th>of</th>
<th>and</th>
<th>a</th>
<th>her</th>
<th>she</th>
<th>his</th>
<th>is</th>
<th>was</th>
<th>in</th>
<th>that</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>75</td>
<td>72</td>
<td>56</td>
<td>52</td>
<td>50</td>
<td>41</td>
<td>36</td>
<td>35</td>
<td>34</td>
<td>34</td>
<td>33</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>had</th>
<th>i</th>
<th>from</th>
<th>you</th>
<th>as</th>
<th>this</th>
<th>mr</th>
<th>for</th>
<th>not</th>
<th>on</th>
<th>be</th>
<th>he</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>28</td>
<td>28</td>
<td>25</td>
<td>23</td>
<td>23</td>
<td>22</td>
<td>21</td>
<td>21</td>
<td>20</td>
<td>18</td>
<td>17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>but</th>
<th>elizabeth</th>
<th>with</th>
<th>him</th>
<th>which</th>
<th>by</th>
<th>when</th>
<th>jane</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>17</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>12</td>
</tr>
</tbody>
</table>
Some counts for “letter” in “Pride and Prejudice”. What do you notice?

<table>
<thead>
<tr>
<th>the</th>
<th>to</th>
<th>of</th>
<th>and</th>
<th>a</th>
<th>her</th>
<th>she</th>
<th>his</th>
<th>is</th>
<th>was</th>
<th>in</th>
<th>that</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>75</td>
<td>72</td>
<td>56</td>
<td>52</td>
<td>50</td>
<td>41</td>
<td>36</td>
<td>35</td>
<td>34</td>
<td>34</td>
<td>33</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>had</th>
<th>i</th>
<th>from</th>
<th>you</th>
<th>as</th>
<th>this</th>
<th>mr</th>
<th>for</th>
<th>not</th>
<th>on</th>
<th>be</th>
<th>he</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>28</td>
<td>28</td>
<td>25</td>
<td>23</td>
<td>23</td>
<td>22</td>
<td>21</td>
<td>21</td>
<td>20</td>
<td>18</td>
<td>17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>but</th>
<th>elizabeth</th>
<th>with</th>
<th>him</th>
<th>which</th>
<th>by</th>
<th>when</th>
<th>jane</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>17</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>12</td>
</tr>
</tbody>
</table>

Almost all the most frequent co-occurring words are function words.
Some words are more informative than others

- Function words co-occur frequently with **all** words
  - That makes them less informative
- They have much higher co-occurrence counts than content words
  - They can “drown out” more informative contexts

**Frequency**

- Just selecting the most frequently occurring bigrams
  \[ \max_{x,y} p(x, y) \quad \max_{x,y} \log(p(x, y) + 1) \]
- A simple POS filter drastically improves the results. Filtering couples of words POS Tagged such as “Aux Noun” deals with results such as “Prime Minister”
Some Statistical Measures

- **Pointwise Mutual Information (Theory of Information)**
  - The PMI tells us the amount of information that is provided by the occurrence of one word about the occurrence of the other word
  
  \[ PMI(x, y) = \log_2 \frac{p(x, y)}{p(x)p(y)} \]

- **Pearson’s Phi-Square Test (Test Hypothesis)**
  - The essence of the test is to compare the observed frequencies with the frequencies expected for independence in a contingency table
  - Try to refute the Null Hypothesis. The Higher the score, the more confidently the Hypothesis Ho can be rejected

  \[ Ho: p(x, y) = p(x)p(y) \]
### Other Measures

There exists many associative measures...

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joint probability</td>
<td>$P(x, y)$</td>
</tr>
<tr>
<td>2</td>
<td>Conditional probability</td>
<td>$P(y</td>
</tr>
<tr>
<td>3</td>
<td>Reverse conditional prob.</td>
<td>$P(x</td>
</tr>
<tr>
<td>4</td>
<td>Pointwise mutual inform.</td>
<td>$\log \frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>5</td>
<td>Mutual dependency (MD)</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>6</td>
<td>Log frequency biased MD</td>
<td>$\log \frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>7</td>
<td>Normalized expectation</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>8</td>
<td>Mutual expectation</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>9</td>
<td>Salience</td>
<td>$\log \frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>10</td>
<td>Pearson’s $r^2$ test</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>11</td>
<td>Fisher’s exact test</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>12</td>
<td>$t$ test</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>13</td>
<td>z score</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>14</td>
<td>Poisson significance measure</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>15</td>
<td>Log likelihood ratio</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>16</td>
<td>Squared log likelihood ratio</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Russel-Rao</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>18</td>
<td>Sokal-Michner</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>19</td>
<td>Rogers-Tanimoto</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>20</td>
<td>Hamann</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>21</td>
<td>Third Sokal-Sneath</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>22</td>
<td>Jaccard</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>23</td>
<td>First Kulczynski</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>24</td>
<td>Second Sokal-Sneath</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
<tr>
<td>25</td>
<td>Second Kulczynski</td>
<td>$\frac{P(x, y)}{P(x)P(y)}$</td>
</tr>
</tbody>
</table>
String level metrics

- String metric is a metric that measures distance between two text strings for approximate string matching or comparison and in fuzzy string searching.

- For example, the strings "Sam" and "Samuel" can be considered to be close.

- A string metric provides a number indicating an algorithm-specific indication of distance.

![Diagram showing string metric with characters 'I N T E N T I O N' and 'E X E C U T I O N' with operations like delete, substitute, and insert indicated.]
String level metrics

- List of string metrics
  - Sørensen–Dice coefficient
  - Block distance or L1 distance or City block distance
  - Jaro–Winkler distance
  - Simple matching coefficient (SMC)
  - Jaccard similarity or Jaccard coefficient or Tanimoto coefficient
  - Tversky index
  - Overlap coefficient
  - Variational distance
  - Hellinger distance or Bhattacharyya distance
  - Information radius (Jensen–Shannon divergence)
  - Skew divergence
  - Confusion probability
  - Tau metric, an approximation of the Kullback–Leibler divergence
  - Fellegi and Sunter metric (SFS)
  - Maximal matches
  - Grammar-based distance
## String level metrics

<table>
<thead>
<tr>
<th>Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hamming distance</td>
<td>&quot;karolin&quot; and &quot;kathrin&quot; has 3.</td>
</tr>
</tbody>
</table>
| Levenshtein distance and Damerau-Levenshtein distance | kitten and sitting have a distance of 3.  
1. kitten → sitten (substitution of "s" for "k")  
2. sitten → sittin (substitution of "i" for "e")  
3. sittin → sitting (insertion of "g" at the end). |
| Jaro-Winkler distance               | $d_j = \frac{1}{3} \left( \frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m - l}{m} \right) = \frac{1}{3} \left( \frac{6}{6} + \frac{6}{6} + \frac{6 - \frac{2}{2}}{6} \right) = 0.944$ |
| Most frequent k characters          | MostFreqKeySimilarity('research', 'seeking', 2) = 2                      |
Word2vec: a state-of-the-art technique

- Word2Vec is an interesting technique that could be used to address this problem
  - Proposed in 2013 (just 5 years ago)
  - Tons of available implementations and ready-to-go calculations
  - Each word is mapped to a vector (not really new idea)
  - The main idea is to predict the relationship between a word and a context using a neural network
Word2Vec - clarifications

- Deep learning ≠ Word embeddings
- Word2Vec is a technique to calculate Word embeddings, but there are many more
- Several public available implementations and already calculated vectors over the Web
- It is cool and works well, but not so easy to train
- Parameter selection is a big deal (as well as computing processing)
Word embeddings evaluation

- They are just vectors, so use them as it.
- Public dataset of analogies.

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>France - Paris</td>
<td>Italy: Rome</td>
<td>Japan: Tokyo</td>
<td>Florida: Tallahassee</td>
</tr>
<tr>
<td></td>
<td>small: larger</td>
<td>cold: colder</td>
<td>quick: quicker</td>
</tr>
<tr>
<td></td>
<td>Baltimore: Maryland</td>
<td>Dallas: Texas</td>
<td>Kona: Hawaii</td>
</tr>
<tr>
<td></td>
<td>Messi: midfielder</td>
<td>Mozart: violnist</td>
<td>Picasso: painter</td>
</tr>
<tr>
<td></td>
<td>Berlusconi: Italy</td>
<td>Merkel: Germany</td>
<td>Koizumi: Japan</td>
</tr>
<tr>
<td></td>
<td>zinc: Zn</td>
<td>gold: Au</td>
<td>uranium: plutonium</td>
</tr>
<tr>
<td></td>
<td>Sarkozy: Nicolas</td>
<td>Putin: Medvedev</td>
<td>Obama: Barack</td>
</tr>
<tr>
<td></td>
<td>Google: Android</td>
<td>IBM: Linux</td>
<td>Apple: iPhone</td>
</tr>
<tr>
<td></td>
<td>Google: Yahoo</td>
<td>IBM: McNealy</td>
<td>Apple: Jobs</td>
</tr>
<tr>
<td></td>
<td>Germany: bratwurst</td>
<td>France: tapas</td>
<td>USA: pizza</td>
</tr>
</tbody>
</table>

- Higher accuracy the better, but don’t forget for what do you want them.

![Graphs showing word embeddings relationships](image)
Word2Vec

Artificial Neural Networks

1-hot encoding

Source Text

The quick brown fox jumps over the lazy dog. →

Training Samples

(the, quick)
(the, brown)

The quick brown fox jumps over the lazy dog. →

(qquick, the)
(qquick, brown)
(qquick, fox)

The quick brown fox jumps over the lazy dog. →

(brown, the)
(brown, quick)
(brown, fox)
(brown, jumps)

The quick brown fox jumps over the lazy dog. →

(fox, quick)
(fox, brown)
(fox, jumps)
(fox, over)

Rome = [1, 0, 0, 0, 0, 0, ..., 0]
Paris = [0, 1, 0, 0, 0, 0, ..., 0]
Italy = [0, 0, 1, 0, 0, 0, ..., 0]
France = [0, 0, 0, 1, 0, 0, ..., 0]
Take away message

- You know the formulas...so, you can code it by your self!

- Depending of the task, preprocessing could be a critical phase. (e.g. orange and oranges)

- Different granularity levels could be use: document, paragraph, sentence, etc.

- Python
  - nltk.metrics.association module

- How to know what is better?
  - Evaluation!!! Many datasets allows to evaluate the performance of existing algorithms (e.g. SemEval datasets, Semantic and Syntactic similarity dataset)