Text Mining

Diego Garrido

Departamento de Ingeniería Industrial
Universidad de Chile

9 de Octubre de 2018

¿Por qué los datos no estructurados son tan importantes?

Las empresas están tomando decisiones solo con el 20% de la información a la que tienen acceso, ya que el 80% de su información es no estructurada y no se puede utilizar completamente. Las compañías han tratado de darle sentido a los datos no estructurados por años, pero solo un 78% afirma que tienen poca o nula información de sus datos no estructurados [1].

Fuentes de datos no estructurados en las empresas:

1. Medios de comunicación social. Ej: mención de una compañía o un producto en redes sociales.
2. Fuentes internas. Ej: Presentaciones, material de marketing y ventas, informes, correo electrónico, etc.
3. Contenido generado por el cliente. Ej: Comentarios en línea, historial de navegación, correos electrónicos a un equipo de soporte e incluso llamadas telefónicas al servicio al cliente.

[1] http://fredrikstenbeck.com/unstructured-data-important/

Embedding
¿Cómo trabajamos con datos no estructurados?

Embedding: Llevar datos a una representación númerica. Los modelos trabajan con representaciones númericas!

En *text mining* existen dos tipos de *Word Embedding*, basados en frecuencia y en predicción.

Algunas *keywords*:

1. Documento (D): se refiere a una observación de tipo texto. Ej: comentario, relato, email, etc.
2. Corpus (C): colección de documentos, corresponde al conjunto de datos.
3. Vocabulario (V): corresponde a todas las palabras únicas encontradas en el corpus que sobrevivieron a un procesamiento.

Count Vector: Matriz cuyas filas son los documentos y las columnas son las palabras del vocabulario V, el valor en cada celda corresponde a la frecuencia de aparición de la palabra en el documento.

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

Doc 1: Buenos días Chile
Doc 2: Buenos días estudiantes buenos

	Buenos	días	Chile	estudiantes
Doc 1	1	1	1	0
Doc 2	2	1	0	1

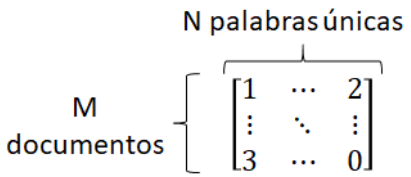


Fig. 1: (a) Matriz término documento transpuesta. (b) Ejemplo de matriz término documento con dos observaciones.

TF-IDF Vector: A partir de la matriz término documento se aplica una transformación, esta transformación consiste en normalizar la frecuencia TF(t,d) de cada palabra t en cada documento d por IDF(t) = $ln(N/f_t)$, donde f_t es las veces que la palabra t ha aparecido al menos una vez en el total de documentos y N es el número de documentos.

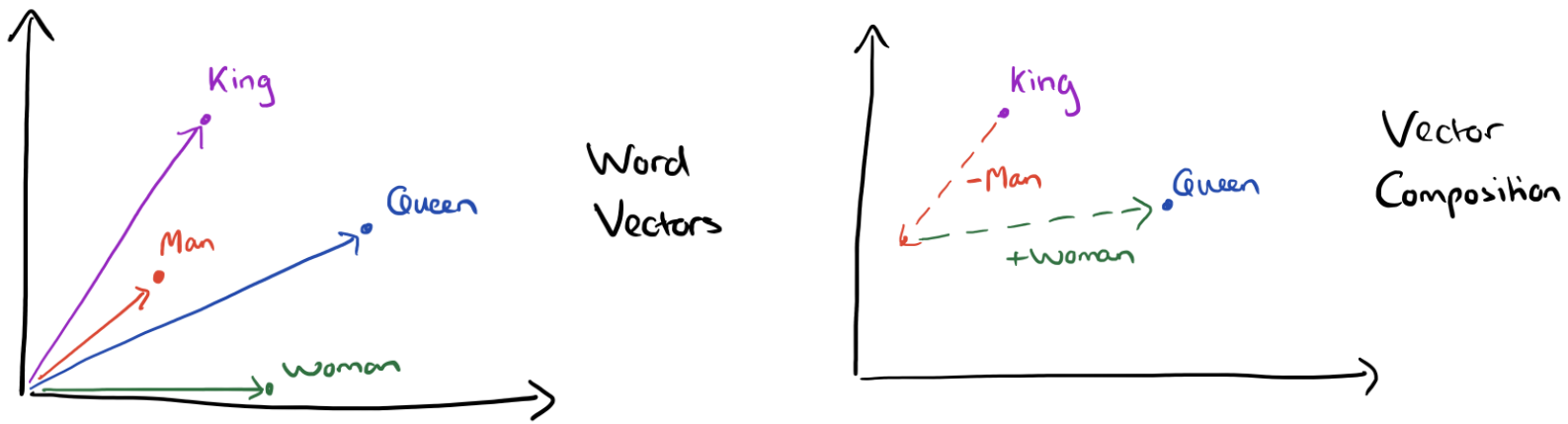
$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

	Buenos	días	Chile	estudiantes
Doc 1	1	1	1	0
Doc 2	2	1	0	1

	Buenos	días	Chile	estudiantes
Doc 1	0	0	1	0
Doc 2	0	0	0	1

Fig. 2: Transformación TF-IDF sobre una matriz término documento.

Los *Words Embedding* basados en predicción son principalmente arquitecturas de redes neuronales que "aprenden" la representación subyacente de las palabra, capturan la "semántica" de las palabras en una representación vectorial densa de baja dimensión y a partir de estos vectores se puede construir representaciones vectoriales de un texto (conjunto de palabras). Ejemplo[2]: Nos gustaría responder la siguiente pregunta, hombre es a rey como mujer es a ?



(a) **(b)**

Fig. 3: Ejemplo de analogías considerando sexo.

[2] https://github.com/uchile-nlp/spanish-word-embeddings/blob/master/examples/Ejemplo_WordVectors.md

Los principales modelos de *Words Embedding* basados en predicción son:

1. **Skip – Gram model** : Se basa en la hipótesis que las palabras que aparecen en contexto similares tienen representaciones vectoriales similares, donde el contexto esta definido por las palabras vecinas, por ejemplo, por una ventana de 10 palabras, 5 palabras a la izquierda y 5 a la derecha. Skip-Gram entrena el contexto contra la palabra, preguntando, "dada esta única palabra, ¿cuáles son las otras palabras que probablemente aparecerán cerca de ella al mismo tiempo?"
2. **CBOW (Continuous Bag of words)**: El modelo CBOW entrena cada palabra en su contexto, preguntando, "dado este conjunto de palabras de contexto, ¿qué palabra faltante es probable que también aparezca al mismo tiempo?"

Una implementación basado en *Skip-Gram model*, entrenado con *Spanish Billion Word Corpus* [3].

[3] https://github.com/uchile-nlp/spanish-word-embeddings

Comparación entre Word Embedding basado en frecuencia vs basado en predicción:

1. En textos no muy largo los *embedding* basados en predicción se alzan por sobre los basados en frecuencia, debida a su alta capacidad semántica.
2. Las principales desventajas de los *embedding* basados en frecuencia es que las representaciones obtenidas son *sparse*, es decir, se obtienen largos vectores con pocos valores no nulos, ya que la cantidad de palabras diferentes fácilmente puede sobrepasar los miles de palabras. Además, no permite relaciones semánticas complejas debido a que las palabras son intercambiables.
3. Las principales desventajas de los *embedding* basados en predicción es que las representaciones obtenidas para textos muy largos no son muy buenas, ya que para obtener una representación vectorial de un texto implica utilizar funciones de agregación sobre vectores de palabras y cuando más largo es el documento la influencia de cada palabra se pierde. Ej: $\vec{V_d} = \frac{\sum_{i=1}^{N} \vec{v_i}}{N}$, para el documento d que tiene N palabras, donde el vector de la palabra $i - ésima$ es $\vec{v_i}$.

Una imágen de un solo canal (escala de grises) puede ser representada por un vector, donde cada columna es un píxel de la imagen y el valor en cada campo corresponde a un valor entre 0-255.

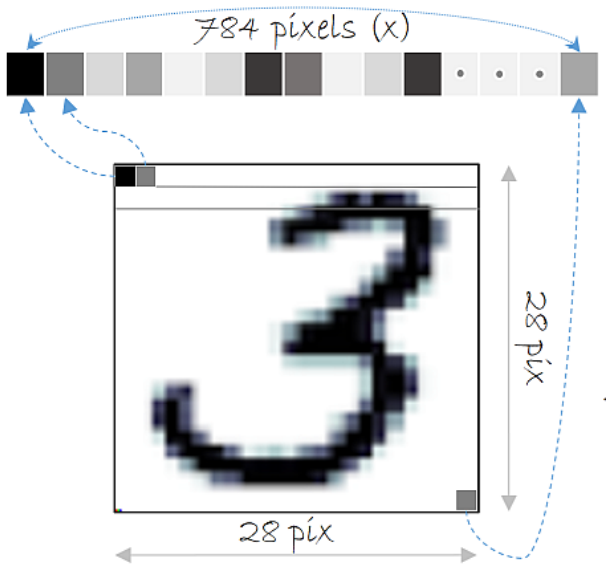


Fig. 4: Representación vectorial de una imagen en escala de grises.

Representación bastante pobre, ya que los píxeles en esta representación son independientes, lo cuál es antinatural, ya que los píxeles cercanos tienen alta dependencia, ¿Sucede lo mismo con la matriz término-documento?

El vocabulario español-latino tiene 88.000 palabras segúb la RAE de las cuáles 20.000 palabras activas y unas 40.000 pasivas se usan normalmente [4]. Nos enfrentamos a un problema de alta dimensionalidad!

Tokenization: dividir el texto en entidades significativa, por ejemplo en palabras unitarias (unigram), pares de palabras (bigram), tres palabras (trigram), etc. Cada entidad representa una columna en la matriz término documento. Pasar de un *string* a una lista de *strings*. Ej:

$$\textit{'Buenos días estudiantes buenos'} = \textit{['Buenos', 'diás', 'estudiantes', 'buenos']}$$

Buenas prácticas de procesamiento:

1. Eliminar caracteres no alfa-numéricos: @_#%$/(?
2. Corrección de Ortografía. Las palabras con mala ortografía suelen tener baja frecuencia.
3. Eliminar números (depende del contexto): números de boletas, indicadores, etc.
4. Identificar palabras claves de ser necesario.

[4] https://elpais.com/diario/2010/11/27/babelia/1290820336_850215.html

¿Cómo podemos reducir el vocabulario?

1. Remover Stop-words, palabras que aportan poca información, por ejemplo: artículos, preposiciones, conectores.
2. Stemming: Llevar las palabra a su raíz gramatical.
3. Lemmatization: Llevar las palabras a su lema.
4. Eliminar palabras con muy baja frecuencia, por ejemplo menor a 10.
5. Eliminar palabras con muy alta frecuencia (stopwords contextuales), por ejemplo aquellas palabras que aparecen en todos los documento. En este caso una transformación TF-IDF genera una columna de ceros, esto implica nula o poca heterogeneidad por ende eliminar.
6. Part of Speech Tagging (POST): etiquetado gramatical. Se suele utilizar en selección de atributos, por ejemplo dejando solo los sustantivos, pues estos son los que tienen la mayor parte de la información en una oración.
7. Usar técnicas de selección de atributos conocidas. Ejemplo: mutual information, chi-square, etc.

Algunas conjugaciones del verbo Jugar, como "Jugaba", "Jugábamos", "Jugué", "Jugasteis", "Jugabais", "Jugado" al ser lematizadas se obtiene "Jugar", en cambio por stemming se obtiene "Jug". Revisar los link en el pie de página para ejemplos en python de procesamiento de texto.[5]

[5] 1) Guide to deal with Text Data with Python 2) Datacamp - Text Analytics for Beginners using NLTK (Python)

El modelamiento de tópicos, es una herramienta estadística que busca encontrar los temas presentes en un conjunto de documentos (corpus), permitiendo organizar, buscar, indexar, explorar y comprender grandes colecciones de documentos.
En este sentido, los temas se pueden definir como "un patrón repetitivo de términos co-currentes en un corpus". Por ejemplo, se tiene el siguiente tópico, representado por sus cuatro palabras más probables, "salud", "médico", "paciente", "hospital", estas palabras sugieren el siguiente nombre para el tema: "Atención médica''.En los modelos convencionales de tópicos:

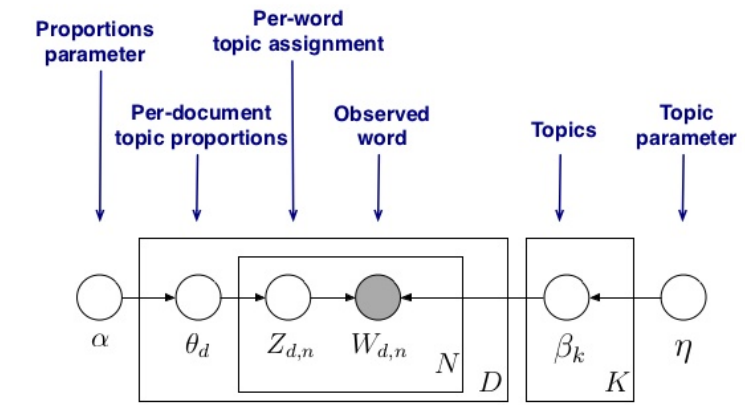
1. Las palabras dentro de un documento y los documentos son tratados como intercambiables.
2. La mayoría de estos fijan el número de tópicos y lo mantienen así a lo largo del corpus completo.

Latent Dirichlet Allocation[6] (LDA)

Sean K tópicos, $\beta_{1:K}$ son distribuciones de probabilidad sobre un vocabulario fijo, dibujadas por una $Dirichlet(\eta)$. Para cada documento d del corpus D se asume que es dibujado por el siguiente proceso generativo:

1. Escoger la mezcla de tópicos θ^d de una distribución sobre un $(K1) - simplex$, tal como una $Dirichlet(\alpha)$.
2. Para cada palabra:
   (a) Escoger la asignación del tópico **z** $Mult(\theta^d)$.
   (b) Escoger una palabra **w** de una $Mult(\beta_z)$.

LDA as a graphical model



$$\prod_{i=1}^{K} p(\beta_i \mid \eta) \prod_{d=1}^{D} p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \beta_{1:K}, z_{d,n}) \right)$$

[6] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan):993–1022.

Las visualizaciones en modelamiento de tópicos nos ayudan a responder tres preguntas:

1. **¿Cuál es el significado de cada tópico?**
2. **¿Cuán predominante es cada tópico?**
3. **¿Cómo se relacionan los tópicos entre sí?**

Sievert, C., Shirley, K. (2014)[7] , desarrollaron una herramienta de visualización para responder estas preguntas. La herramienta a través de una visualización espacial responde la pregunta 2 y 3. Además para responder la pregunta 1 incorporan un gráfico de barras a la derecha del gráfico espacial que muestra las palabras más relevantes del tópico seleccionado dado un parámetro λ entre 0 y 1, entonces, la relevancia de la palabra w en el tópico k dado λ esta dada a través de la siguiente formula:

$$r(w, k|\lambda) = \lambda log(\phi_{k,w}) + (1 - \lambda) log(\frac{\phi_{k,w}}{p_w}), \lambda \in [0, 1]$$

Donde $\phi_{k,w}$ es la probabilidad de que el término w sea generado por el tópico k, p_w es la probabilidad de el término w en el corpus.

`Click aquí: LDAvis example`

[7] Sievert, C., Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. In Proceedings of the workshop on interactive language learning, visualization, and interfaces (pp. 63-70).