



UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL CÓRDOBA

# Diplomatura en Data Science Aplicada

# Agenda

- Análisis de sentimientos últimas ideas
- Feature engineering en NLP
-

# Sentiment Analysis

últimas nociones

# Sentiment Analysis



También conocido como:

- Extracción de opinión.
- Minería de opiniones.
- Análisis de subjetividad.
- Minería de sentimientos



Tipos de valoración para una opinión:

- Positiva
- Negativa
- Neutra



Queda mucho por hacer:

- El lenguaje humano es complejo y se encuentra evolucionando.
- Enseñar a una máquina que analice los matices gramaticales, slangs, errores de escrituras comunes. O enseñar a nuestro modelo que entienda como un determinado contexto afecta o no es complejo.
- Es complejo generalizar un modelo para todos los idiomas.

# Sentiment Analysis - Desafíos en general

Cada fuente de datos presenta un tratamiento particular. Y la exploración de los mismos o el conocimiento a priori son el lugar donde dar inicio.

- Scraping de páginas
  - Tags html, xml, templates, etc.
- Twitter u otras redes
  - texto mal formado, url, hashtags, menciones, etc.
- Emoticones
  - como los mapeamos a información, siempre nos sirven?
- Dependiendo el idioma
  - Cómo manejar negaciones, contracciones, etc en el Inglés?
    - u, r , cause
  - El Español depende mucho de su región, que estrategia vamos a tomar?
- Teléfonos, fechas, localizaciones, etc cada cual posee formatos distintos.

# Sentiment Analysis - Trabajando con reseñas

Al ser un reflejo de una opinión, es difícil que un modelo pueda entender los tratamientos gramaticales complejos.

- Entender el contexto actual de lo que se está diciendo, es sumamente complejo para una computadora.
- Las personas se tornan creativas a la hora de usar un emoticón, y es complejo entender el significado → 🤪
- Si usamos, TODO EN MAYÚSCULA, otra vez estamos frente a una dicotomía.

# Sentiment Analysis - Problemas

## Elipsis e ironía

- *@LovNaty Tu vida ha parido a un grandísimo hijo de la gran p... , un **maravilloso** hombre!!*. (NEG)
- En la huerta cultivamos el tomate, y el pepino también. (Se omite el verbo “cultivar”).

"I don't know how to say this," she said, looking down.

"You mean he's..."

"Yes, he's gone. I'm sorry."

## Doble negación

- ***No** es que ahora **no** sea feliz, **pero** antes lo era **más*** (NEG)

Es lindo pero la confección regular



Natalia  
@NRivarola

No pagó el rescate del perro.  
De eso murió.  
Fin.

[#Carmel](#)

[Translate Tweet](#)

4:23 PM · Nov 9, 2020 from Córdoba, Argentina



# Sentiment Analysis - Problemas

## Polaridad



Bien, no tiene pinta de que fuera a durar pero ya veré

Bien aunque tiene poco volumen y la sopapa no se mantiene pegada

Es increíble. La única contra es que la batería no dura lo que esperaba. Por el uso que le doy necesita 2 cargas completas al día



# Sentiment Analysis - Problemas

Dan Jurafsky



## Thwarted Expectations and Ordering Effects

- “This film should be **brilliant**. It sounds like a **great** plot, the actors are **first grade**, and the supporting cast is **good** as well, and Stallone is attempting to deliver a good performance. However, it **can’t hold up**.”
- Well as usual Keanu Reeves is nothing special, but surprisingly, the **very talented** Laurence Fishbourne is **not so good** either, I was surprised.

# Sentiment Analysis - Trabajando con reseñas

Dan Jurafsky



## Extracting Features for Sentiment Classification

- How to handle negation
  - I **didn't** like this movie
  - vs
  - I really like this movie
- Which words to use?
  - Only adjectives
  - All words
    - All words turns out to work better, at least on this data

# Sentiment Analysis - Trabajando con reseñas

Dan Jurafsky



## Negation

Das, Sanjiv and Mike Chen. 2001. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In Proceedings of the Asia Pacific Finance Association Annual Conference (APFA).  
Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP-2002, 79—86.

Add NOT\_ to every word between negation and following punctuation:

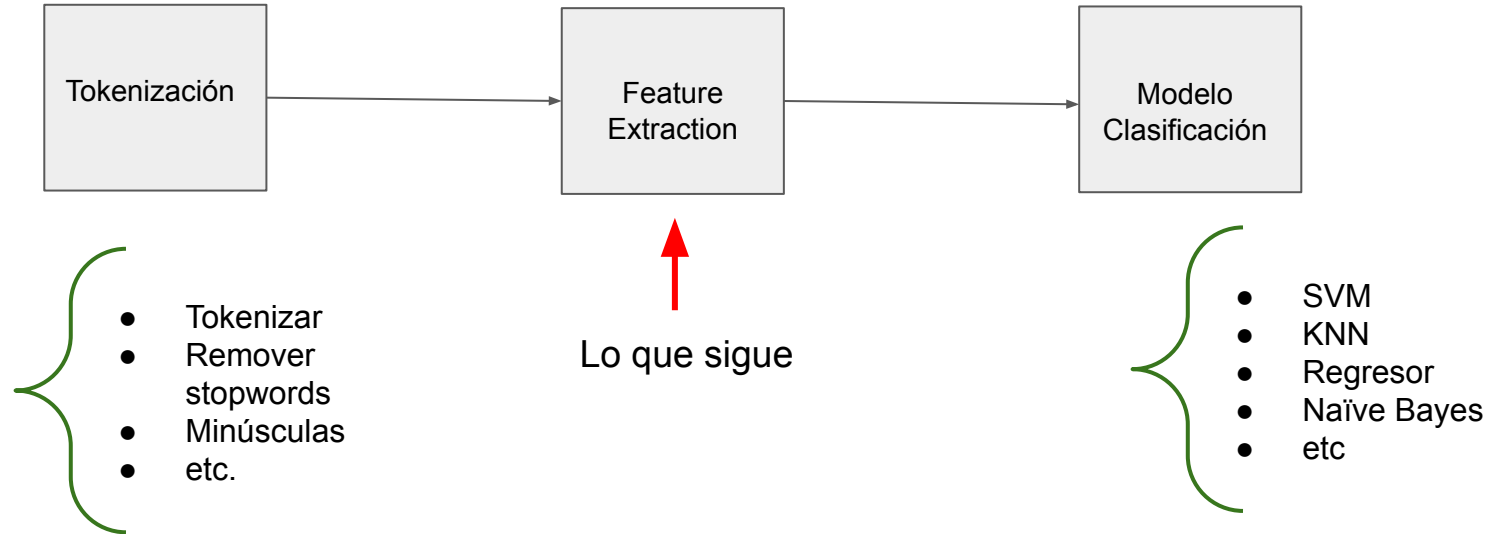
didn't like this movie , but I



didn't NOT\_like NOT\_this NOT\_movie but I

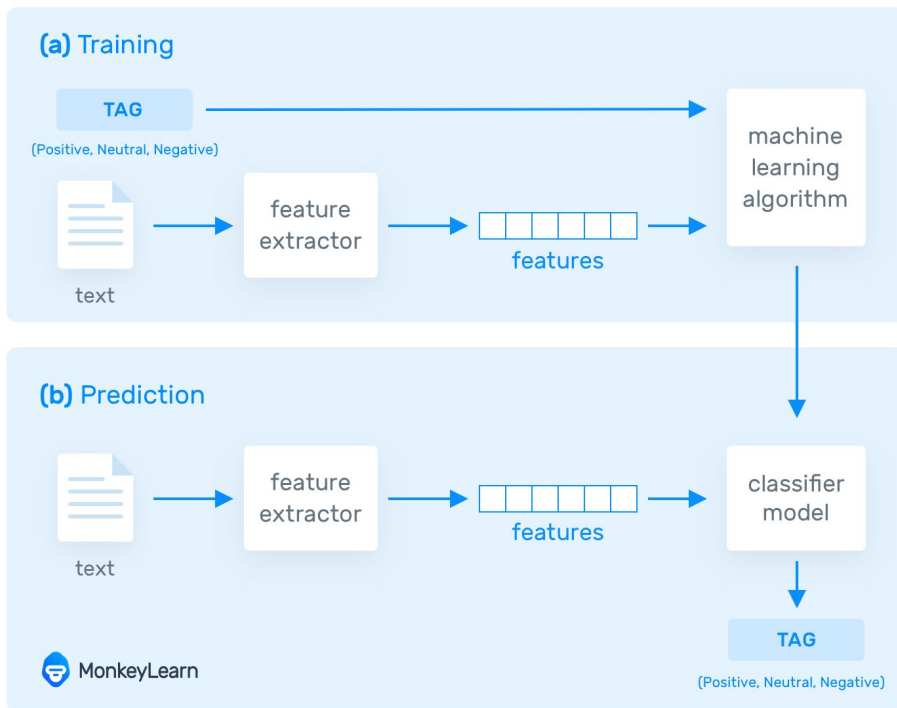
# Sentiment Analysis - Cerrando

Pipeline para llegar a un modelo base:



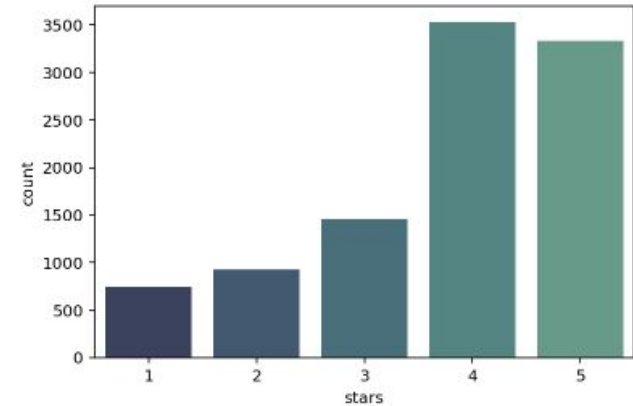
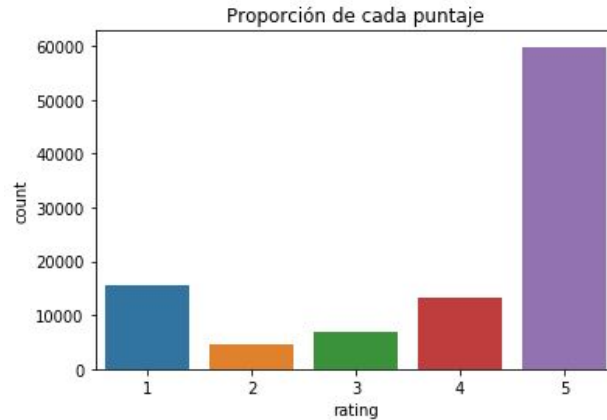
# Sentiment Analysis - Esto no es todo.

## How Does Sentiment Analysis Work?



¿cómo resolvemos la falta de etiquetas?

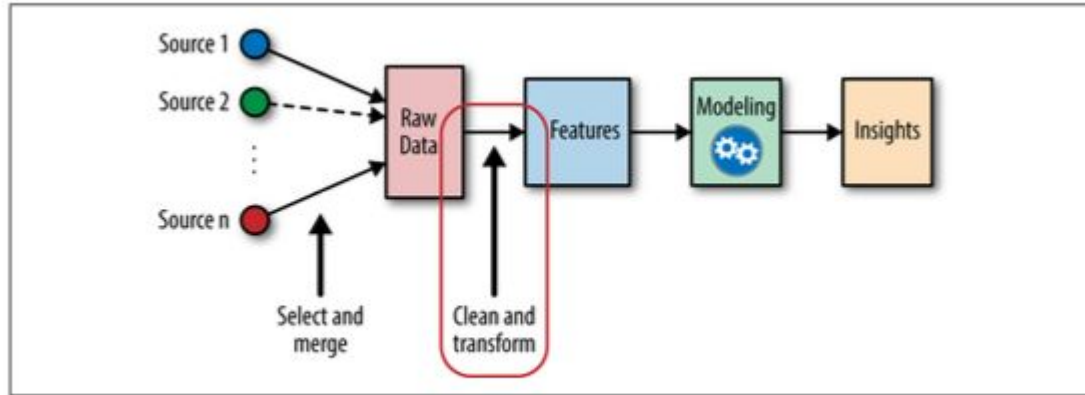
# Sentiment Analysis - Esto no es todo.



# Feature Engineering

# Feature Engineering

“Feature engineering is the process of formulating the most appropriate features given the data, the model, and the task.”



*Figure 1-2. The place of feature engineering in the machine learning workflow*



# Feature Engineering - Features

¿Qué es una feature?

Representación numérica sobre distintos datos, crudos sin ser procesados.

Tipos de features

Numéricas, texto, imágenes, etc.

**Trabajaremos con features:**

- numéricas
- texto

¿Cuántas features?

**No sé**, pero debemos siempre buscar aquellas que sean relevantes. Si contamos con pocas el modelo no va a lograr predecir correctamente, y si tenemos demasiadas será pesado, complejo para entrenar.

*“The right features are relevant to the task at hand and should be easy for the model to ingest”*

# Features de texto - Vectorización

“Es el proceso de codificar el texto como un valor numérico, para crear feature vectors que serán interpretados por los modelos de machine learning,”

Lo que ya vimos:

- A **pre-processing** step to make the texts cleaner and easier to process

Ahora vamos a ver

- A **vectorization** step to transform these texts into numerical vectors.



“Todos sabemos que el túnel que pasa bajo las vías en la estación Flores es una de las entradas del infierno. Cierta noche de otoño, el ruso Salzman...”

# Vectorización de datos - Bag of Words (BoW)

“bag-of-words, is the simplest representation based on word count statistics”

Todos sabemos que el túnel que pasa  
bajo las vías en la estación Flores es  
una de las entradas del infierno



Todos	1
el	1
vías	1
infierno	1
casa	0
es	1
otras	0
la	0
las	1

# Vectorización de datos - Bag of N-Words

Extensión de BoW, y volvemos al concepto de N-Gram definido por Markov.

Todos sabemos que el túnel que pasa bajo las vías en la estación Flores es una de las entradas del infierno



## Algunos bigramas:

- vías en
- flores es
- el tunel



Teóricamente con  $k$  palabras únicas van haber  $k^2$  2-grama únicos.

unigram

C O L D

C O L D

C O L D

C O L D

bigram

C O L D

C O L D

C O L D

trigram

C O L D

C O L D

n-gram ( $n = 4$ )

C O L D

# Features numéricas- Intuiciones

“Puede venir de una variedad de fuentes: geolocalización, precios de productos, medidas de telemetría, etc. A pesar de estar ya en valores numéricos no significa que no se necesite realizar un tratamiento para hacer más inteligible al modelo.”

Algunas preguntas a responder:

- Las magnitudes importan, son correctas?  
Valores extremos, que hacemos con ellos?
- Nos importa el valor absoluto, o si son positivos o negativos?
- Estos valores sería mejor escalarlos? Para evitar valores extremos, o magnitudes muy variables.

# Features de texto - Vectorización

## Apply CountVectorizer

```
In [17]: 1 from sklearn.feature_extraction.text import CountVectorizer
2
3 count_vect = CountVectorizer(analyzer=clean_text)
4 X_counts = count_vect.fit_transform(data['body_text'])
5 print(X_counts.shape)
6 print(count_vect.get_feature_names())
```

## Apply CountVectorizer (N-Grams)

```
In [20]: 1 from sklearn.feature_extraction.text import CountVectorizer
2
3 ngram_vect = CountVectorizer(ngram_range=(2,2),analyzer=clean_text) # It applies only bigram vectorizer
4 X_counts = ngram_vect.fit_transform(data['body_text'])
5 print(X_counts.shape)
6 print(ngram_vect.get_feature_names())
```

## Apply TfidfVectorizer


```
In [22]: 1 from sklearn.feature_extraction.text import TfidfVectorizer
2
3 tfidf_vect = TfidfVectorizer(analyzer=clean_text)
4 X_tfidf = tfidf_vect.fit_transform(data['body_text'])
5 print(X_tfidf.shape)
6 print(tfidf_vect.get_feature_names())
```

---

# Feature Engineering

## Checklist: Data Cleaning

Get printable version [here](#). For more detailed instructions on how to implement the different tasks in this checklist, see [Data Cleaning](#). Note that this checklist is best displayed in Chrome, Firefox, Safari or any other modern browser.



Project name: \_\_\_\_\_

Country: \_\_\_\_\_

District: \_\_\_\_\_

Year, Month and/or Day: \_\_\_\_\_

### 1. Before data cleaning: Importing the data

	Initials	#No	Checklist Item
[...]		1.1	Check for importing issues such as broken lines when importing .csv files
[...]		1.2	Make sure you have unique IDs
[...]		1.3	De-identify all data and save in a new .dta file
[...]		1.4	Never make any changes to the raw data

### 2. Important steps for data cleaning

	Initials	#No	Checklist Item
[...]		2.1	Label variables, don't use special characters
[...]		2.2	Recode and label missing values: your data set should not have observations with -777, -88 or -9 values, for example
[...]		2.3	Encode variables: all categorical variables should be saved as labeled numeric variables, no strings
[...]		2.4	Don't change variable names from questionnaire, except for nested repeat groups and reshaped roster data
[...]		2.5	Check sample representativeness of age, gender, urban/rural, region and religion
[...]		2.6	Check administrative data such as date, time, interviewer variables included
[...]		2.7	Test variables consistency
[...]		2.8	Identify and document outliers
[...]		2.9	Compress dataset so it is saved in the most efficient format
[...]		2.10	Save cleaned data set with an informative name. Avoid saving in a very recent Stata version

### 3. Optional steps in data cleaning

	Initials	#No	Checklist Item
[...]		3.1	Order variables – unique ID always first, then same order as questionnaire
[...]		3.2	Drop variables that only make sense for questionnaire review (duration, notes, calculates)
[...]		3.3	Rename roster variables
[...]		3.4	Categorize variables listed as "others"
[...]		3.5	Add metadata as notes: original survey question, relevance, constraints, etc

The checklist is edited through [GitHub](#). This checklist corresponds to the file with the name **chk\_datacleaning.js**. To read a simple step by step guide on how to edit the checklist, see this [documentation](#):  
<https://github.com/worldbank/DIMEwiki/tree/master/Topics/Checklists>.

Referencia para arrancar a trabajar o no olvidarnos en el futuro

[https://dimewiki.worldbank.org/wiki/Checklist:\\_Data\\_Cleaning](https://dimewiki.worldbank.org/wiki/Checklist:_Data_Cleaning)