comillas.edu

Universidad Pontificia Comillas
ICAI

Analítica Social y de la Web
Social and Web Analytics

3.1 Procesado de datos

Javier Ruiz de Ojeda

Curso 2023-24
Segundo semestre

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

Introducción

Introducción

Por qué monitorizar nuestra marca

- Una vez hemos visto las técnicas fundamentales para la recopilación de datos sociales, el siguiente paso natural es el de utilizar estos datos para el análisis de la actividad de una marca, que podría ser la nuestra propia o la de nuestro cliente, y hacer frente así a cuestiones naturales sobre la popularidad, aceptación y/o sentimientos que se pueden desprender de las comunicaciones entre la marca y sus seguidores.
- A día de hoy, cualquier marca comercial utiliza (múltiples) redes sociales para comunicarse con sus potenciales consumidores, bien sea informando sobre nuevos productos, servicios, noticias de la compañía de relevancia, colaboraciones con otras marcas/personas de interés, etc.

Introducción
Por qué monitorizar nuestra marca

- Este flujo de comunicación, como ya hemos podido anticipar en el apartado anterior, es abrumador, pudiendo perdernos en la infinidad de contenidos de lo que realmente se está hablando (y cómo) en ellos.
- El objetivo principal de este apartado es precisamente el de mostrar el procedimiento genérico de colección-procesamiento-análisis (también conocido como ***pipeline***), a partir del cual podremos extraer conclusiones para resolver nuestras preguntas sobre la actividad y percepción de la marca y sus productos. Para ello, usaremos marcas conocidas que tengan actividad considerable en redes sociales(al menos Facebook).

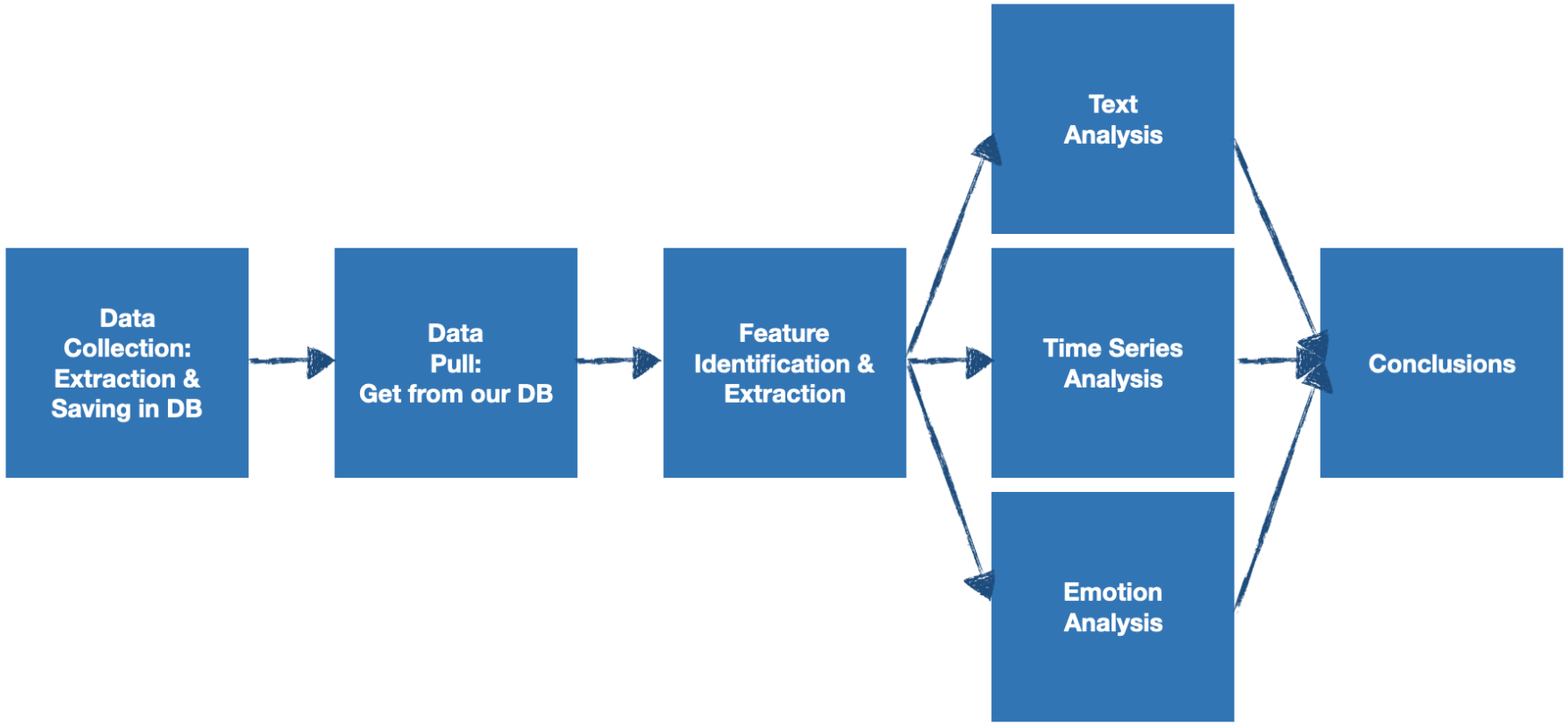
Introducción
Apartados del capítulo

Así, en este capítulo cubriremos los siguientes puntos:

- Extracción de datos sociales de la página oficial de Facebook de la marca (posts y comentarios)
- Limpieza/preprocesado de los datos sociales
- Procesado para la extracción de palabras clave (keywords), bigramas, y hashtags
- Procesado de comentarios para el análisis de sentimiento
- Análisis de los resultados

Introducción

Fases del análisis



Introducción
Planificación de la tarea

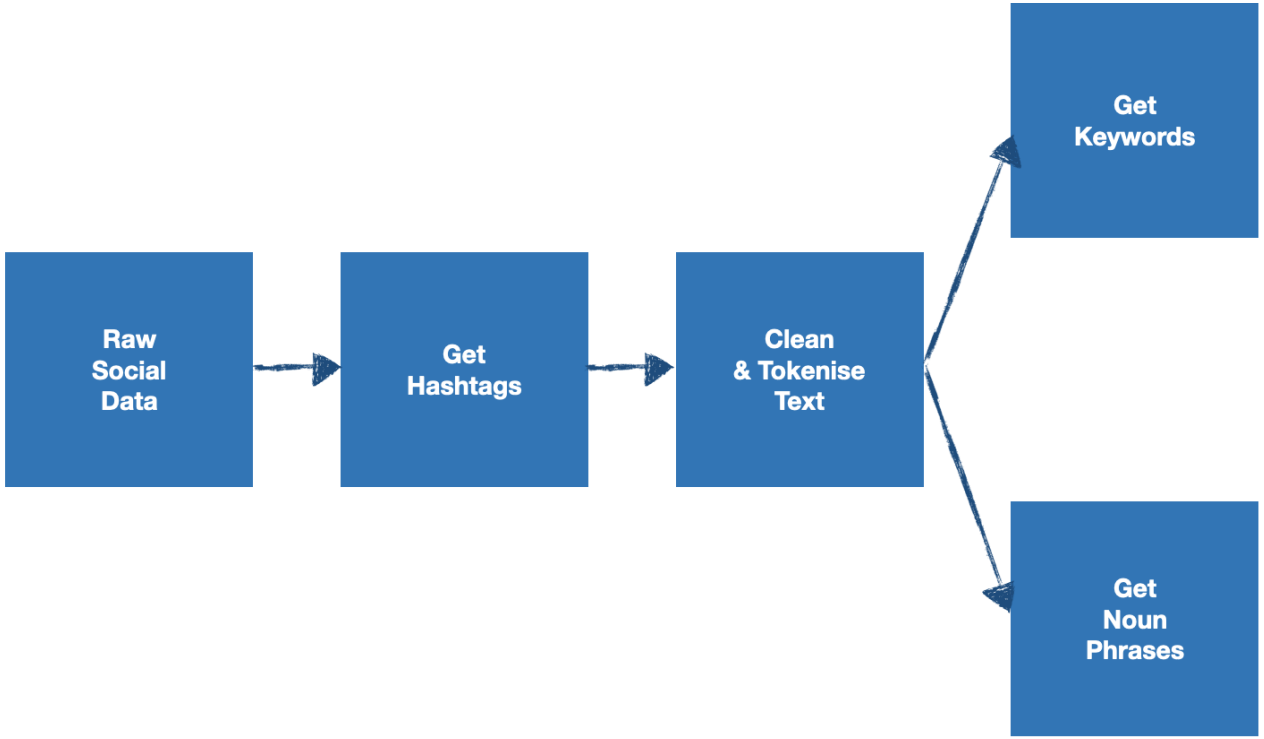
- Como siempre, lo primero que tenemos que hacer es determinar cuáles son las preguntas que queremos responder. Obviamente, la determinación de estas preguntas no será algo estático, sino que será modificado iterativamente según vayamos avanzando en cada una de las etapas del proyecto (metodología *agile*).
- En esta tarea vamos a intentar dar respuesta a las siguientes preguntas "tentativas":
  - ¿Qué está publicando la marca seleccionada en Facebook?
  - ¿Cómo están reaccionando los seguidores de la marca a dichas publicaciones? (likes, shares, comentarios, retweets)
  - ¿Qué se comenta sobre la marca en Facebook?
  - ¿Qué emociones están surgiendo de las publicaciones?

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

Funciones de procesado

Funciones de procesado
Procesado

Una vez hemos descargado los datos que consideramos oportunos a nuestra base de datos, procederemos a su procesamiento para extraer la información fundamental que nos permitirá responder a las cuestiones planteadas anteriormente. En esencia, los pasos que vamos a seguir son los siguientes:



Funciones de procesado
Obtener #hashtags


```
import re

def extract_hashtags(text: str):
  hashtags = list(set(re.findall(
      pattern=r"#(\w+)",
      string=text
  )))
  return hashtags
```

Funciones de procesado

Limpieza, estandarización y tokenización


```
import re
import nltk
def text_preprocessing(text: str):
  # cleans white spaces and punctuation, and converts text to lower
  c_text = re.sub(
      pattern=r"[^\w\s]",
      repl="",
      string=text.lower().strip()
  )
  # tokenize words:
  tokens = nltk.word_tokenize(c_text)
  return tokens
```

Funciones de procesado

Etiquetado de tokens


```
1  import nltk
2
3  tokens = nltk.word_tokenize("Can you please buy me an Arizona Ice Tea? It's $0.99.")
4
5  print("Parts of Speech: ", nltk.pos_tag(tokens))
```

```
[('Can', 'MD'), ('you', 'PRP'), ('please', 'VB'), ('buy', 'VB'),
('me', 'PRP'), ('an', 'DT'), ('Arizona', 'NNP'), ('Ice', 'NNP'),
('Tea', 'NNP'), ('?', '.'), ('It', 'PRP'), ("'s", 'VBZ'), ('$',
'$'), ('0.99', 'CD'), ('.', '.')]
```

Funciones de procesado
Etiquetado de tokens

- CC coordinating conjunction
- CD cardinal digit
- DT determiner
- EX existential there (like: “there is” … think of it like “there exists”)
- FW foreign word
- IN preposition/subordinating conjunction
- JJ adjective ‘big’
- JJR adjective, comparative ‘bigger’
- JJS adjective, superlative ‘biggest’
- LS list marker 1)
- MD modal could, will
- NN noun, singular ‘desk’
- NNS noun plural ‘desks’
- NNP proper noun, singular ‘Harrison’
- NNPS proper noun, plural ‘Americans’
- PDT predeterminer ‘all the kids’
- POS possessive ending parent’s
- PRP personal pronoun I, he, she
- PRP$ possessive pronoun my, his, hers

Funciones de procesado
Etiquetado de tokens

- RB adverb very, silently,
- RBR adverb, comparative better
- RBS adverb, superlative best
- RP particle give up
- TO, to go ‘to’ the store.
- UH interjection, errrrrrrrm
- VB verb, base form take
- VBD verb, past tense took
- VBG verb, gerund/present participle taking
- VBN verb, past participle taken
- VBP verb, sing. present, non-3d take
- VBZ verb, 3rd person sing. present takes
- WDT wh-determiner which
- WP wh-pronoun who, what
- WP$ possessive wh-pronoun whose
- WRB wh-abverb where, when

Funciones de procesado
Etiquetado de tokens


```
import nltk

def tag_tokens(text):
  # Get the part-of-speech of a word in a sentence:
  pos = nltk.pos_tag(text)
  return pos
```

Funciones de procesado
Extracción de sustantivos


```
def extract_keywords(tagged_tokens, types="all", types_list=("NN", "JJ", "VP")):
  if types == "all":
      tag_types = types_list
  elif types == "nouns":
      tag_types = "NN"
  elif types == "verbs":
      tag_types = "VB"
  elif types == "adjectives":
      tag_types = "JJ"
  else:
      tag_types = types_list

  keywords = [
      t[0] for t in tagged_tokens if t[1].startswith(tag_types)
  ]
  return keywords
```

Funciones de procesado

Extracción de sintagmas nominales


```
import nltk
def extract_noun_phrases(tagged_tokens):
    # Optional determiner, and multiple adjts and nouns
    grammar = "NP: {<DT>?<JJ>*<NN>}"
    cp = nltk.RegexpParser(grammar)
    tree = cp.parse(tagged_tokens)
    result = []

    def is_noun(token):
        return token.label() == "NP"
    for subtree in tree.subtrees(filter = is_noun):
        leaves = subtree.leaves()
        if len(leaves) > 1:
            outputs = " ".join([
                t[0] for t in leaves
            ])
            result += [outputs]
    return result
```

Funciones de procesado
Beneficios de la extracción

- Es fiable y consistente porque emplea criterios predefinidos
- Permite análisis en tiempo real sobre la información obtenida de páginas web, redes sociales, encuestas, reseñas, etc.
- Es escalable y permite el análisis automatizado de grandes cantidades de información textual

Funciones de procesado
Métodos de extracción

- **Metodología lingüística**: se basa en el texto y en las palabras que contiene (“partes del lenguaje”, relaciones entre palabras, marcadores de discurso, información semántica, ocurrencia de palabras…)
- **Metodología gráfica**: se basa en transformar el texto en un grafo y analizar los vértices y aristas que componen las relaciones.
- **Metodología estadística**: en vez de requerir datos de entrenamiento, se basa en una aproximación estadística por ocurrencia de las palabras (colocación, frecuencia, combinación, etc.)
- **Metodología híbrida**: combina dos o más metodologías.

Funciones de procesado
Librerías de extracción

- **RAKE**
- **YAKE**
- **PKE**
- **KeyBERT**
- **MiltiRake**
- **TextRank**

Funciones de procesado

Integración en un pipeline


```
def processing_pipeline(df: DataFrame, msg_col: str):
    df["hastags"] = df.apply(
        lambda x: extract_hashtags(x[msg_col]),
        axis=1)
    df["preprocessed"] = df.apply(
        lambda x: preprocess_and_tokenize(x[msg_col]),
        axis=1)
    df["tagged"] = df.apply(
        lambda x: tag_tokens(x["preprocessed"]),
        axis=1)
    df["keywords"] = df.apply(
        lambda x: extract_keywords(x["tagged"]),
        axis=1)
    df["noun_phrases"] = df.apply(
        lambda x: extract_noun_phrases(x["tagged"]),
        axis=1)

    return df
```

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

¡Muchas gracias!