Clasificación automática de textos

Rocío Leal Díaz (roledi301@Gmail.com)
Minería de Datos Clínicos
Ingeniería de la Salud

Índice

¿Qué es la clasificación automática de textos?

“

Una acción desarrollada por un sistema sobre un conjunto de textos para ordenarlos en categorías o clases según sus características.

”

¿Qué necesitamos?

- Biblioteca scikit-learn

- Dataset supervisado: Ohsumed
  - Subconjunto de Medline:

    ‘Recoge las referencias bibliográficas de los artículos publicados en casi 5.000 revistas del área biomédica desde los años sesenta del siglo pasado.’

- Esquema
  1. Preprocesamiento de datos
  2. Entrenamiento de un clasificador
  3. Evaluación del clasificador

Train_set y test_set

- División del dataset en dos particiones: train y test

sklearn.model_selection.train_test_split

- Método que realiza la partición de un conjunto de datos

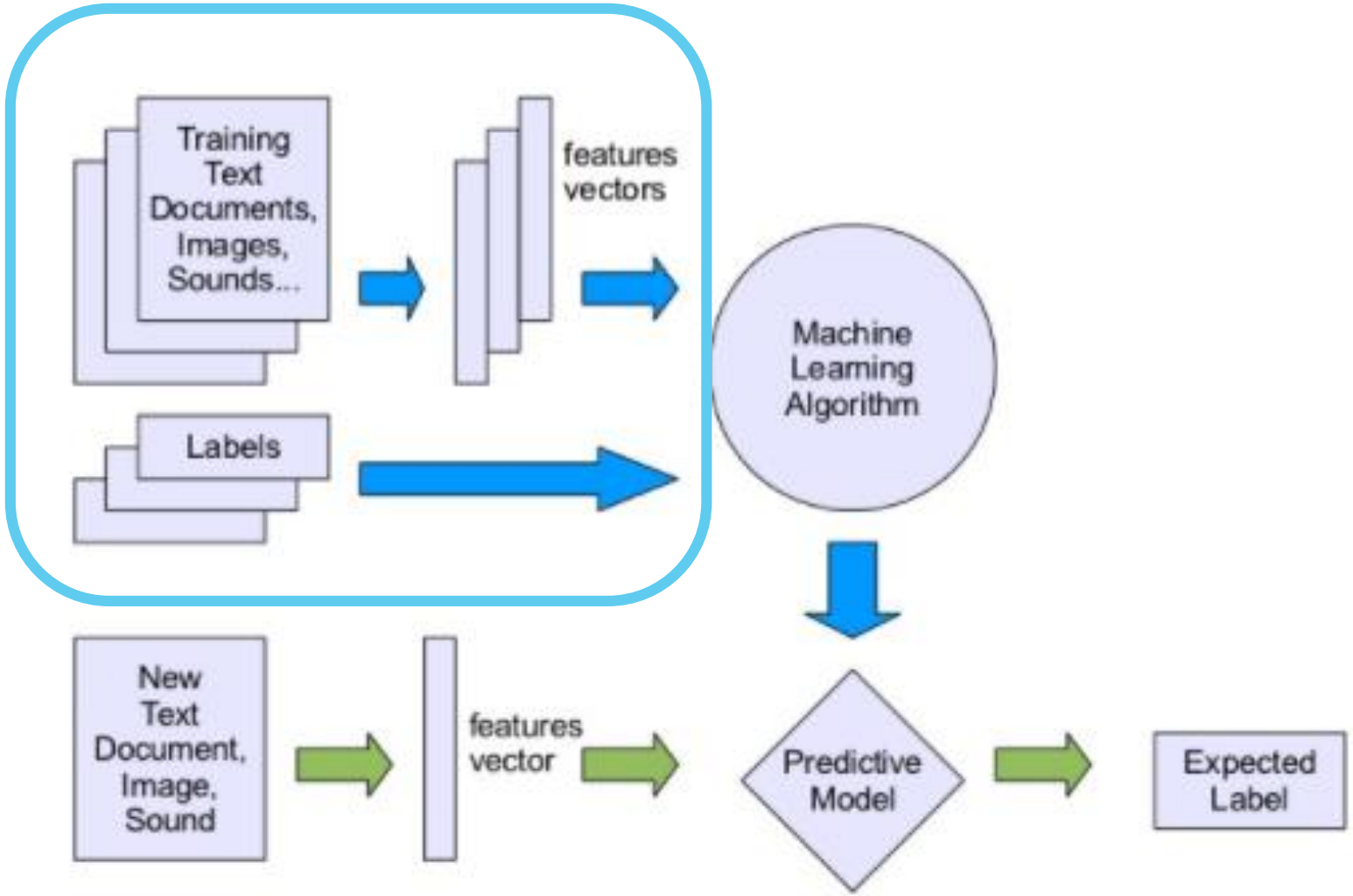
```
X_train, X_test, y_train, y_test = train_test_split(documents, labels, test_size=0.5, random_state=0)
```

Datos de Ohsumed:

- 56979 archivos
- 23 categorías

```
Tamano de X_train:
 28489
Tamano de X_test:
 28490
```

1


Training Text Documents, Images, Sounds...
features vectors
Labels
Machine Learning Algorithm
New Text Document, Image, Sound
features vector
Predictive Model
Expected Label

Extracting features from text files

- Transformación de los textos en vectores numéricos
- Bolsa de palabras (Bags of words)

  ‘Estructura que contiene todas las palabras que aparecen en los archivos junto con el número de veces que aparece en el mismo’

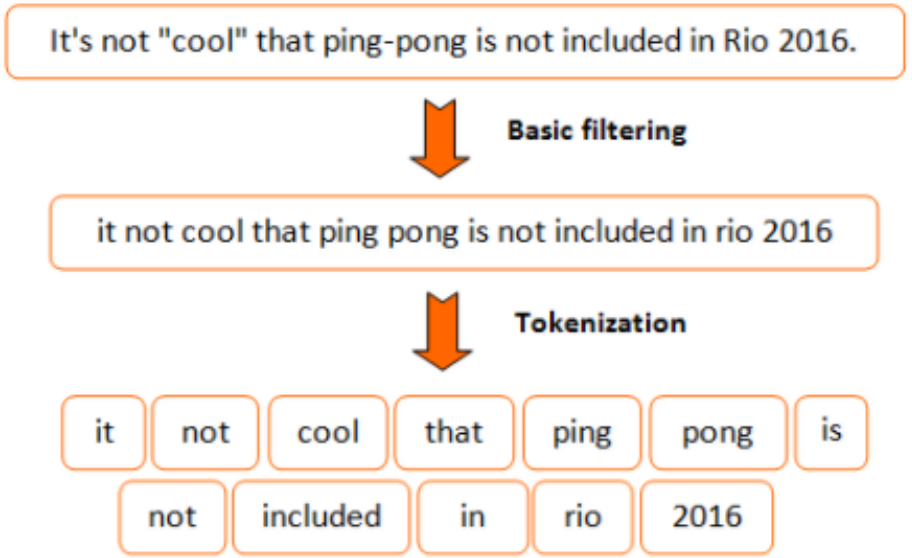
  - Proceso de creación:

    1. Tokenización de los párrafos en palabras
    2. Asignación de un ‘id’ a cada palabra
    3. Contar la frecuencia de la palabra por documento

`X[i,j]`

Tokenizing text

- Engloba el preprocesamiento de datos, separación en palabras y filtrado por ‘stopwords’



Tokenizing text

- Clase

sklearn.feature_extraction.text.CountVectorizer

class sklearn.feature_extraction.text. **CountVectorizer** (*input=u'content', encoding=u'utf-8', decode_error=u'strict', strip_accents=None, lowercase=True, preprocessor=None, tokenizer=None, stop_words=None, token_pattern=u'(?u)\b\w\w+\b', ngram_range=(1, 1), analyzer=u'word', max_df=1.0, min_df=1, max_features=None, vocabulary=None, binary=False, dtype=<type 'numpy.int64'>*) [source]

Convert a collection of text documents to a matrix of token counts

- Método

fit_transform (*raw_documents, y=None*) [source]

Learn the vocabulary dictionary and return term-document matrix.

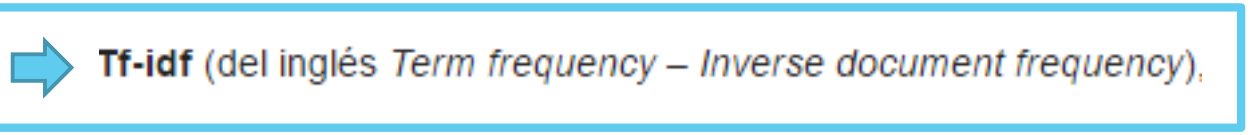
Tokenizing text

```
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
```

```
(0, 14440)    1
(0, 40534)    1
(0, 14614)    1
(0, 3554)     1
(0, 14622)    1
(0, 44421)    1
(0, 33749)    1
```

Longitud de documentos

- Los documentos de tamaño grande tendrán una frecuencia más alta que los documentos de tamaño pequeño
  - Solución: Normalizar el vector de frecuencia



  - Medida que expresa cómo es de importante una palabra para un documento de un colección
  - Aumenta proporcionalmente según el número de veces que una palabra aparece en el documento, pero este valor se compensada con la frecuencia de la palabra en la colección de documentos

Tf-idf

- Producto de: frecuencia de término (tf) y frecuencia inversa de documento (idf)
  - Frecuencia de término tf(t, d): número de veces que el término *t* aparece en el documento *d*

$$\mathrm{tf}(t,d) = \frac{\mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}}$$

  - Frecuencia inversa de documento idf(t,d): medida que indica si un término es común o no. Se hace el logaritmo del cociente del número total de documentos entre el número de documentos que contienen el término

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

Tf-idf

- Clase

sklearn.feature_extraction.text.TfidfTransformer

class sklearn.feature_extraction.text. **TfidfTransformer** (*norm=u'l2', use_idf=True, smooth_idf=True, sublinear_tf=False*) [source]

Transform a count matrix to a normalized tf or tf-idf representation

- Método

fit_transform (*X, y=None, **fit_params*) [source]

Fit to data, then transform it.

Tf-idf

```
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
```

```
(0, 16320)    0.0350454824398
(0, 34241)    0.0547368096077
(0, 43130)    0.286039708618
(0, 40531)    0.0667046683809
(0, 4920)     0.100595315317
(0, 23397)    0.045044344369
(0, 34434)    0.0846073012149
(0, 40904)    0.180763982049
(0, 48344)    0.0989412668666
```

Innovación

- Clase

sklearn.feature_extraction.text.TfidfVectorizer

class sklearn.feature_extraction.text. **TfidfVectorizer** (*input=u'content', encoding=u'utf-8', decode_error=u'strict', strip_accents=None, lowercase=True, preprocessor=None, tokenizer=None, analyzer=u'word', stop_words=None, token_pattern=u'(?u)\b\w\w+\b', ngram_range=(1, 1), max_df=1.0, min_df=1, max_features=None, vocabulary=None, binary=False, dtype=<type 'numpy.int64'>, norm=u'l2', use_idf=True, smooth_idf=True, sublinear_tf=False*) [source]

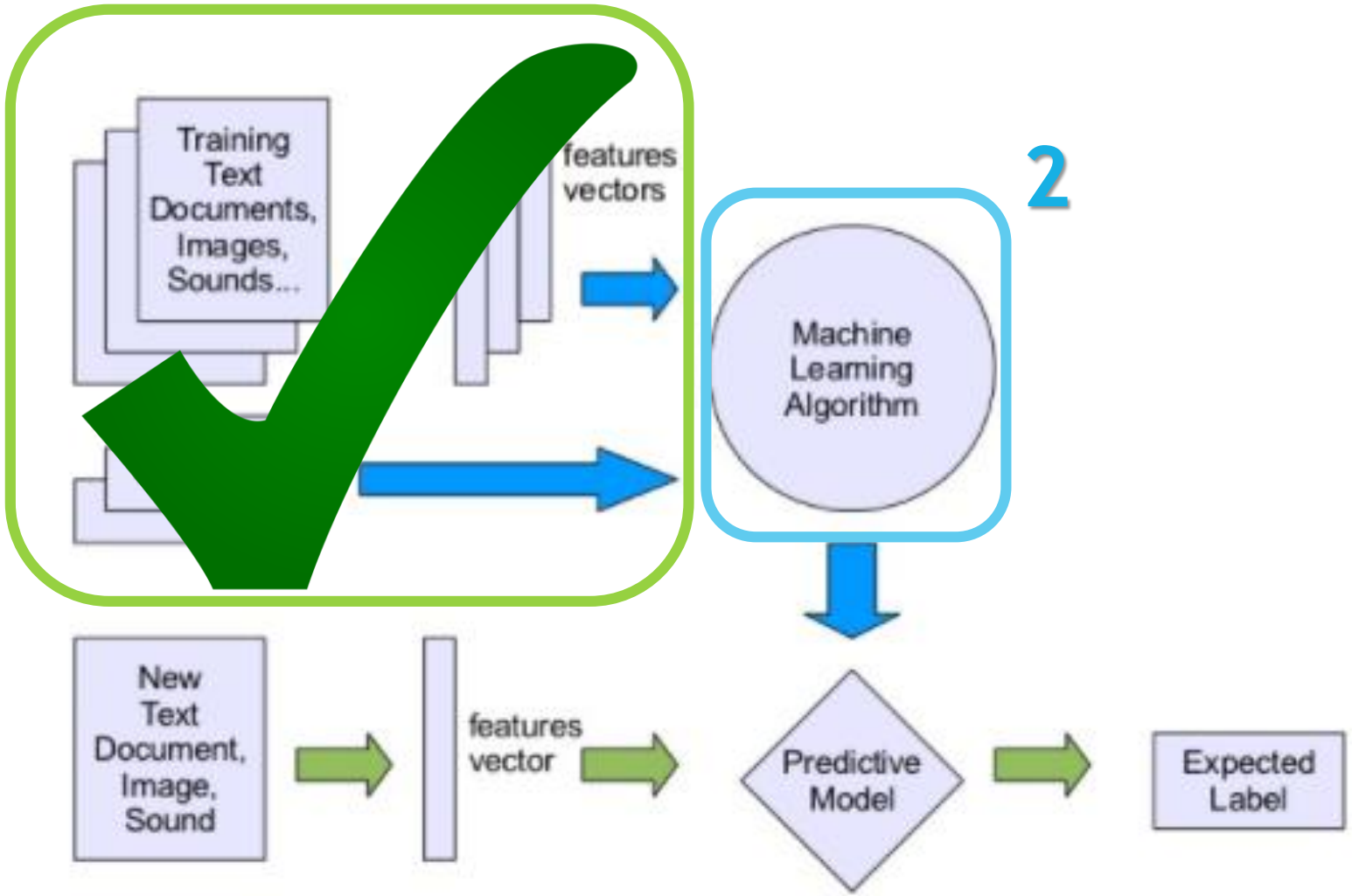
Convert a collection of raw documents to a matrix of TF-IDF features.

Equivalent to CountVectorizer followed by TfidfTransformer.

Innovación

```
stop = set(stopwords.words('english'))
vectorizer = TfidfVectorizer(min_df=1,stop_words=stop)
X_train_vectorizer = vectorizer.fit_transform(X_train)
```

```
(0, 16287)    0.0371283341842
(0, 43048)    0.303039854274
(0, 40450)    0.0706691147296
(0, 23347)    0.0477214566446
(0, 40822)    0.191507294713
(0, 45902)    0.189257368561
(0, 7962)     0.141684728291
(0, 38985)    0.208693645498
(0, 21710)    0.133459757068
```


Training Text Documents, Images, Sounds...
features vectors
2
Machine Learning Algorithm
New Text Document, Image, Sound
features vector
Predictive Model
Expected Label

Entrenamiento del clasificador

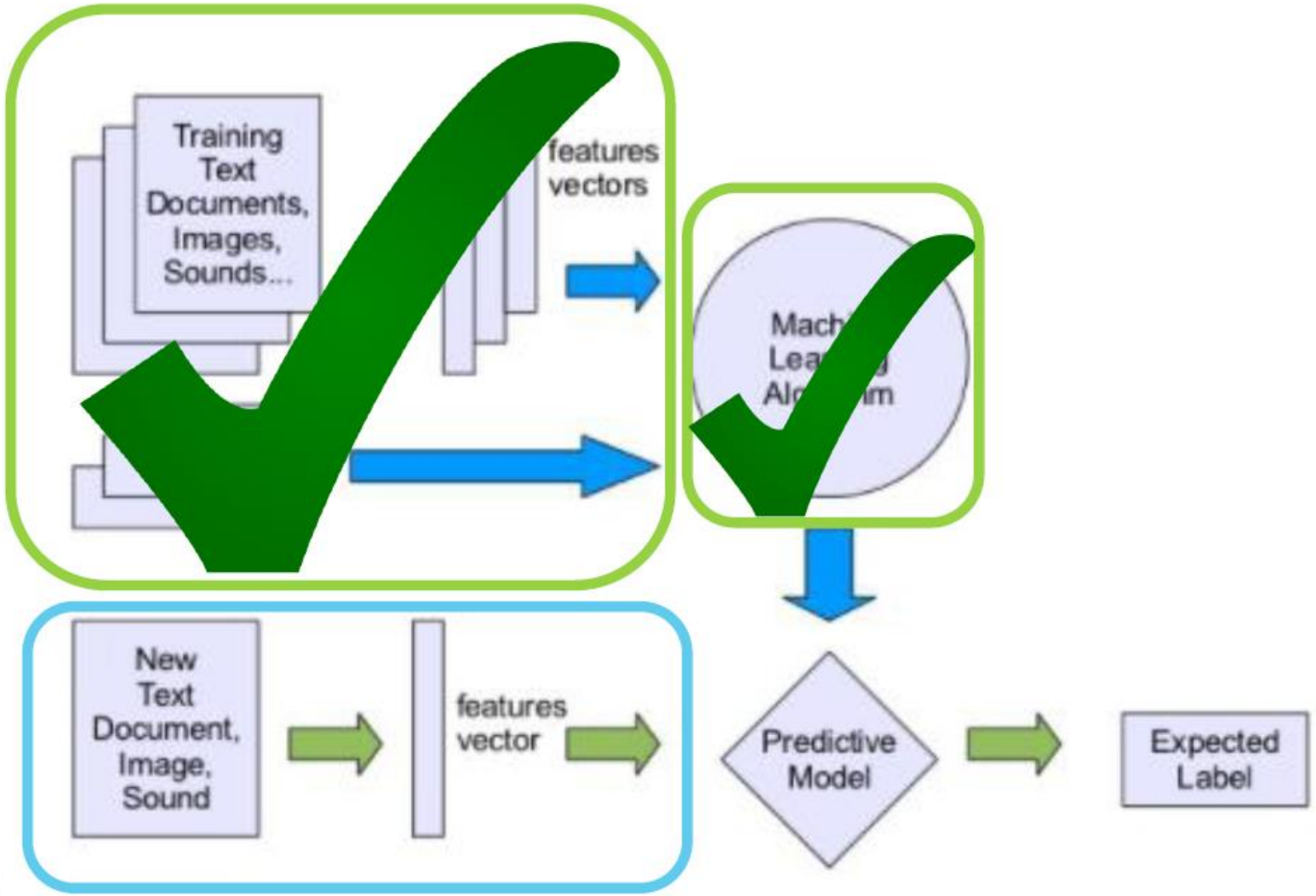
- Gracias al vector de características, se puede entrenar un clasificador para predecir la categoría de un archivo.

1. Definimos el clasificador

```
clfLR = LogisticRegression()
```

2. Entrenamos al clasificador

```
clfLR.fit(X_train_vectorizer, y_train)
```


Training
Text
Documents,
Images,
Sounds...
features
vectors
Mach
Lea
Alg
m
New
Text
Document,
Image,
Sound
features
vector
Predictive
Model
Expected
Label

Evaluación del clasificador

- Obtener el vector de características del Test_set

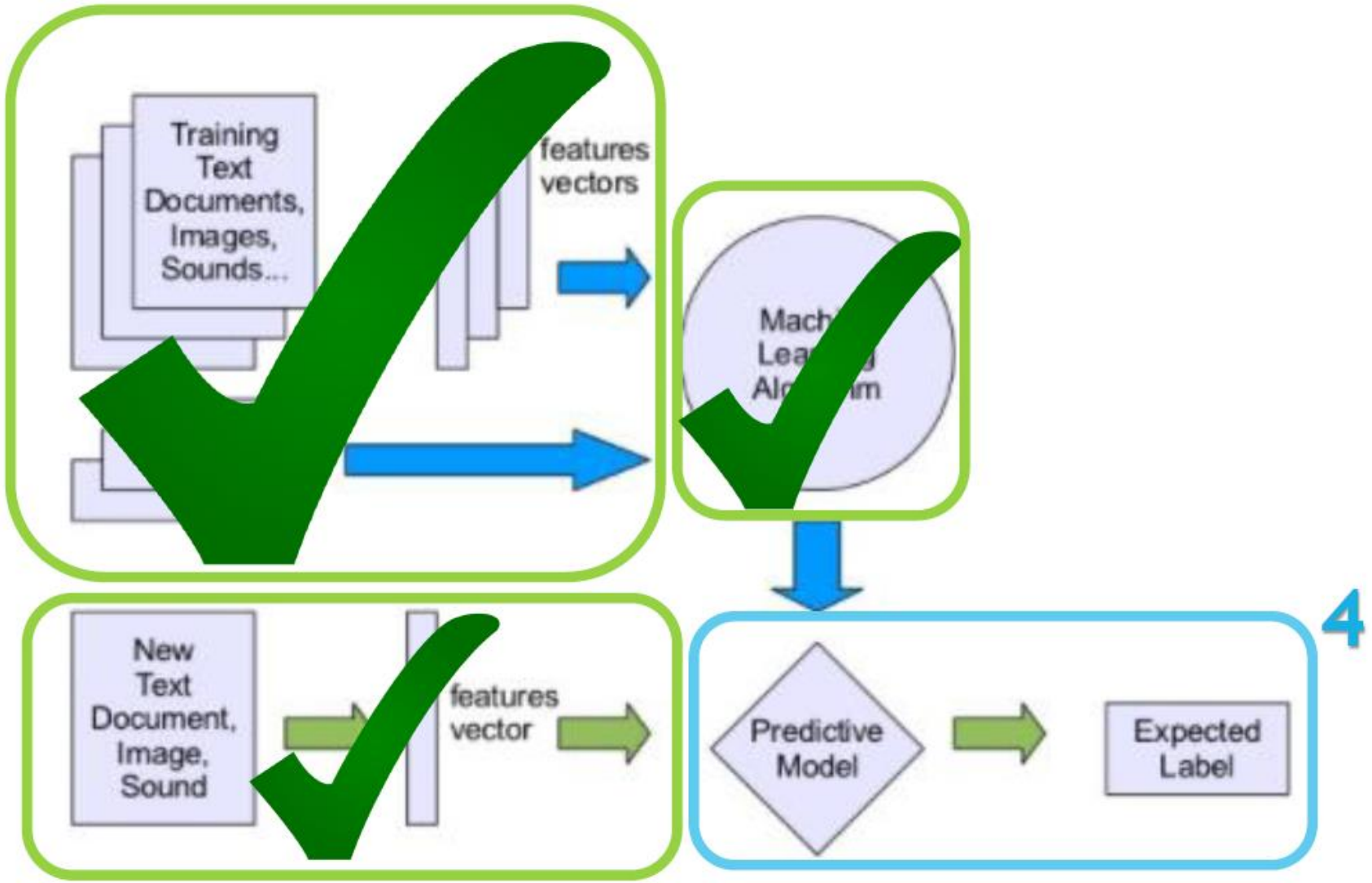
```
#-Tokenización y frecuencia
X_test_counts = count_vect.transform(X_test)

#-Normalización
#tfidf
X_test_tfidf = tfidf_transformer.transform(X_test_counts)

#-Combinación de tokenización, frecuencia y normalización
X_test_vectorizer = vectorizer.transform(X_test)
```

- Predicción de etiqueta

```
clMNB.predict(X_test_vectorizer)
```


Training Text Documents, Images, Sounds...
features vectors
Mach
Lea
Alg
m
New Text Document, Image, Sound
features vector
Predictive Model
Expected Label
4

Evaluación del clasificador

- Resultado

```
Texto:
 The development of medical education on alcohol- and drug-related problems at
 Medical education on alcohol- and drug-related problems at the University of
 An undergraduate core curriculum, developed in the early 1970s, is offered in
 The undergraduate program is rated highly by students; since 1978, 3024 have
 Residency training started in 1974 and is available through electives lasting
 To date, 370 residents have completed one of these electives; 129 have comple
 Despite the progress, there is still a need to improve and expand the undergr
 The goals should be to ensure that, as far as possible, all medical graduates

Categoria: C21

Texto:
 Severity of Helicobacter-induced gastric injury correlates with gastric juice
 We postulated that ammonia produced by Helicobacter pylori may contribute to
 This hypothesis was evaluated in Helicobacter-positive patients with chronic
 Gastric urea and ammonia were measured, and the severity of gastritis was eva
 High gastric ammonia and low urea in Helicobacter-positive patients, and the
 There was a significant correlation between gastric ammonia and interstitial
 Eradication of Helicobacter pylori was associated with a decrease of ammonia
 The significant correlation between the severity of gastric inflammation and

Categoria: C06

Texto:
 Differential alpha-fetoprotein lectin binding in hepatocellular carcinoma. Di
 The reactivity of serum alpha-fetoprotein (AFP) from 20 patients with hepatoc
 The reactivity with Concanavalin A (Con A) was also significantly greater for
 The reactivity with lentil lectin permitted distinction between those with HC
 Evaluation of the alterations by lectin binding methodology may be useful in

Categoria: C04
```

Comparación de la precisión de los clasificadores

Resultado para el clasificador MultinomialNB usando TfidfTransformer

Testing score: 25.2%

Resultado para el clasificador MultinomialNB usando TfidfVectorizer

Testing score: 26.5%

Resultado para el clasificador LogisticRegression usando TfidfVectorizer

Testing score: 40.3%

Resultado para el clasificador DecisionTreeClassifier usando TfidfVectorizer

Testing score: 22.6%

Resultado para el clasificador Perceptron usando TfidfVectorizer

Testing score: 27.2%

Bibliografía

- http://scikit-learn.org/stable/index.html (16/05/2017 - 21:30)
- Tema: Primer clasificador – MDC
- http://www.joragupra.com/2016/03/clasificacion-automatica-de-textos.html

  (16/05/2017 - 16:30)

Muchas Gracias!

Rocío Leal Díaz (roledi301@Gmail.com)

Minería de Datos Clínicos

Ingeniería de la Salud