# Analizador de Textos Documentation

***Release 0.0.6***

**Datos Argentina**

**Dec 13, 2017**

# Contents


**1 Analizador de Textos** **3**
- 1.1 Instalación . . . 3
- 1.2 Uso . . . 4
- 1.3 Tests . . . 5
- 1.4 Créditos . . . 6
- 1.5 Contacto . . . 6

**2 History** **7**
- 2.1 0.0.6 (2017-09-25) . . . 7
- 2.2 0.0.5 (2017-07-14) . . . 7
- 2.3 0.0.4 (2016-11-25) . . . 7
- 2.4 0.0.1 (2016-11-22) . . . 7

**3 Indices y tablas** **9**

Contents:

CHAPTER 1

# Analizador de Textos

Paquete en python para análisis, clasificación y recuperación de textos, utilizado por el equipo de Datos Argentina.


- *Instalación*
  - *Dependencias*
  - *Desde pypi*
  - *Para desarrollo*
- *Uso*
  - *Búsqueda de textos similares*
  - *Clasificación de textos*
- *Tests*
- *Créditos*
- *Contacto*
- Licencia: MIT license


## 1.1 Instalación

### 1.1.1 Dependencias

`textar` usa `pandas`, `numpy`, `scikit-learn` y `scipy`. Para que funcionen, se requiere instalar algunas dependencias no pythonicas:

- En Ubuntu: `sudo apt-get install libblas-dev liblapack-dev libatlas-base-dev gfortran`

### 1.1.2 Desde pypi

```
pip install textar
```

### 1.1.3 Para desarrollo

```
git clone https://www.github.com/datosgobar/textar.git
cd path/to/textar
pip install -e .
```

Cualquier cambio en el código está disponible en el entorno virtual donde fue instalado de esta manera.

## 1.2 Uso

### 1.2.1 Búsqueda de textos similares

```
from textar import TextClassifier

tc = TextClassifier(
    texts=[
        "El árbol del edificio moderno tiene manzanas",
        "El árbol más chico tiene muchas mandarinas naranjas, y está cerca del␣
↪monumento antiguo",
        "El edificio más antiguo tiene muchos cuadros caros porque era de un␣
↪multimillonario",
        "El edificio más moderno tiene muchas programadoras que comen manzanas␣
↪durante el almuerzo grupal"
    ],
    ids=map(str, range(4))
)

ids, distancias, palabras_comunes = tc.get_similar(
    example="Me encontré muchas manzanas en el edificio",
    max_similars=4
)

print ids
['0', '3', '2', '1']

print distancias
[0.92781458944579009, 1.0595805639371083, 1.1756638126839645, 1.3206413200640157]

print palabras_comunes
[[u'edificio', u'manzanas'], [u'edificio', u'muchas', u'manzanas'], [u'edificio', u
↪'muchas'], [u'muchas']]
```

### 1.2.2 Clasificación de textos

```
from textar import TextClassifier

tc = TextClassifier(
    texts=[
```

```
        "Para hacer una pizza hace falta harina, tomate, queso y jamón",
        "Para hacer unas empanadas necesitamos tapas de empanadas, tomate, jamón y␣
↪queso",
        "Para hacer un daiquiri necesitamos ron, una fruta y un poco de limón",
        "Para hacer un cuba libre necesitamos coca, ron y un poco de limón",
        "Para hacer una torta de naranja se necesita harina, huevos, leche, ralladura␣
↪de naranja y polvo de hornear",
        "Para hacer un lemon pie se necesita crema, ralladura de limón, huevos, leche␣
↪y harina"
    ],
    ids=map(str, range(6))
)

# entrena un clasificador
tc.make_classifier(
    name="recetas_classifier",
    ids=map(str, range(6)),
    labels=["Comida", "Comida", "Trago", "Trago", "Postre", "Postre"]
)

labels_considerados, puntajes = tc.classify(
    classifier_name="recetas_classifier",
    examples=[
        "Para hacer un bizcochuelo de chocolate se necesita harina, huevos, leche y␣
↪chocolate negro",
        "Para hacer un sanguche de miga necesitamos pan, jamón y queso"
    ]
)

print labels_considerados
array(['Comida', 'Postre', 'Trago'], dtype='|S6')

print puntajes
array([[-3.52493526,  5.85536809, -6.05497008],
       [ 2.801027  , -6.55619473, -3.39598721]])

# el primer ejemplo es un postre
print sorted(zip(puntajes[0], labels_considerados), reverse=True)
[(5.8553680868184079, 'Postre'),
 (-3.5249352611212568, 'Comida'),
 (-6.0549700786502845, 'Trago')]

# el segundo ejemplo es una comida
print sorted(zip(puntajes[1], labels_considerados), reverse=True)
[(2.8010269985828997, 'Comida'),
 (-3.3959872063363505, 'Trago'),
 (-6.5561947275785393, 'Postre')]
```

## 1.3 Tests

Los tests sólo se pueden correr habiendo clonado el repo. Luego instalar las dependencias de desarrollo:

```
pip install -r requirements_dev.txt
```

y correr los tests:

```
nosetests
```

## 1.4 Créditos

- Victor Lavrenko nos ayudó a entender el problema con sus explicaciones en youtube: https://www.youtube.com/user/victorlavrenko

## 1.5 Contacto

Te invitamos a [crearnos un issue](https://github.com/datosgobar/textar/issues/new?title=Encontré un bug en textar) en caso de que encuentres algún bug o tengas feedback de alguna parte de `textar`.

Para todo lo demás, podés mandarnos tu comentario o consulta a datos@modernizacion.gob.ar.

CHAPTER 2

# History

## 2.1 0.0.6 (2017-09-25)

- Arreglo de bugs en las palabras destacadas de los resultados sugeridos.

## 2.2 0.0.5 (2017-07-14)

- Mejoras en la forma en que se seleccionan las palabras destacadas de la busqueda
- Correcciones a los tests correspondientes

## 2.3 0.0.4 (2016-11-25)

- Correcciones a los tests
- Revisión de la documentación

## 2.4 0.0.1 (2016-11-22)

- First release on PyPI.

CHAPTER 3

# Indices y tablas

- genindex
- modindex
- search