Proyecto: Analitica de textos

Caso: Elegibilidad de pacientes para ensayos clinicos

Camilo Salinas

Para este proyecto, se escogio utilizar el clasificador de Naive Bayes para determinar a los pacientes en si son elegibles o si no. Antes de poder entrenar el modelo se debe realizar un preprocesamiento exaustivo en donde se utilice el modelo de Bag of Words para la vectorización del texto, y despues realizar la lematización del mismo.

0. Importación de librerias

```
import pandas as pd
pd.set_option('display.max_columns', 25)
pd.set_option('display.max_rows', 50)
import numpy as np
np.random.seed(3301)
import pandas as pd

# Preprocesamiento de datos
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
#para hacer balanceo de los features
from imblearn.over_sampling import SMOTE
# Para realizar la separaciond el conjunto de aprendizaje en entrenamiento y test
from sklearn.model_selection import train_test_split
# Para evaluar el modelo
from sklearn.metrics import confusion_matrix, classification_report, precision_score, recall_score, f1_score, accuracy_score
from sklearn.metrics import plot_confusion_matrix
# Para busqueda de hiperparametros
from sklearn.model_selection import GridSearchCV
# Para la validación cruzada
from sklearn.model_selection import KFold
#Librerias para la visualizacion
import matplotlib.pyplot as plt
#Seaborn
import seaborn as sns

import re

from sklearn.preprocessing import FunctionTransformer

import nltk

nltk.download('stopwords')
from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer


pd.set_option('display.max_colwidth', None)  # or 199

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.linear_model import LogisticRegression

from sklearn.pipeline import Pipeline
from sklearn.pipeline import make_pipeline
from sklearn.metrics import mean_squared_error as mse

from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import MultinomialNB


%matplotlib inline
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/rembrandtsx/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

1. Preprocesamiento

Primero se deben cargar los datos y realizar un perfilamiento de estos, una vez se sabe el estado y pureza de los datos se pueden empezar a limpiar siguiendo lo establecido en el diccionario de datos

```
# Importe
df_eleg = pd.read_csv('../Datos/ElegibilidadEstudiantes/clinical_trials_on_cancer_data_clasificacion.csv', sep=',', encoding='utf-8'
df_eleg.head()
```

	label	study_and_condition
0	__label__0	study interventions are Saracatinib . recurrent verrucous carcinoma of the larynx diagnosis and patients must agree to use adequate birth control for the duration of study participation and for at least eight weeks after discontinuation of study drug
1	__label__1	study interventions are Stem cell transplantation . hodgkin lymphoma diagnosis and history of congenital hematologic immunologic or metabolic disorder which in the estimation of the pi poses prohibitive risk to the recipient
2	__label__0	study interventions are Lenograstim . recurrent adult diffuse mixed cell lymphoma diagnosis and creatinine clearance crcl greater than fifty ml per minute all tests must be performed within twenty-eight days prior to registration
3	__label__0	study interventions are Doxorubicin . stage iii diffuse large cell lymphoma diagnosis and stages ii bulky disease defined as mass size of more than ten cm stage iii or iv ann_arbor staging patients with stage and stage ii non bulky disease are excluded from this study
4	__label__1	study interventions are Poly I-C . prostate cancer diagnosis and unresolved iraes following prior biological therapy except that stable and managed iraes may be acceptable hypothyroidism or hypopituitarism on appropriate replacement

Vemos si existen datos nulos en el dataset:

```
df_eleg.isna().sum()
```

```
label                0
study_and_condition    0
dtype: int64
```

Los datos no tienen nulos pero estan presentados de maneras distintas. Algunos comienzan con comillas, otros son espacios. Debemos lograr entradas similares. Adicionalmente la informacion importante del paciente esta ubicada despues del primer punto. Adicionalmente vemos que los labels son categoricos y debemos volvernos numericos. Aunque un label encoder haria esta tarea con facilidad, queremos conservar la categoria implicita que ya traen.

```
def preprocessor(df):
    df= df.replace('"', '')
    df = df.str.strip(' ')
    df = df.str.split('.').str[1]
    return df
print(df_eleg.describe())

def encoder(df):
    df.loc[df['label'] == '__label__0', 'label'] = 0
    df.loc[df['label'] == '__label__1', 'label'] = 1
encoder(df_eleg)
df_eleg
```

```
             label  \
count        12000
unique           2
top     __label__0
freq          6000

                                                                                study_and_condition
count                                                                                         12000
unique                                                                                        11987
top     study interventions are Fludarabine . anaplastic large cell lymphoma diagnosis and donor
freq                                                                                              2
```

	label	study_and_condition

	label	study_and_condition
0	0	study interventions are Saracatinib . recurrent verrucous carcinoma of the larynx diagnosis and patients must agree to use adequate birth control for the duration of study participation and for at least eight weeks after discontinuation of study drug
1	1	study interventions are Stem cell transplantation . hodgkin lymphoma diagnosis and history of congenital hematologic immunologic or metabolic disorder which in the estimation of the pi poses prohibitive risk to the recipient
2	0	study interventions are Lenograstim . recurrent adult diffuse mixed cell lymphoma diagnosis and creatinine clearance crcl greater than fifty ml per minute all tests must be performed within twenty-eight days prior to registration
3	0	study interventions are Doxorubicin . stage iii diffuse large cell lymphoma diagnosis and stages ii bulky disease defined as mass size of more than ten cm stage iii or iv ann_arbor staging patients with stage and stage ii non bulky disease are excluded from this study
4	1	study interventions are Poly I-C . prostate cancer diagnosis and unresolved iraes following prior biological therapy except that stable and managed iraes may be acceptable hypothyroidism or hypopituitarism on appropriate replacement
...	...	...
11995	0	study interventions are Prednisolone hemisuccinate . recurrent childhood large cell lymphoma diagnosis and no known hypersensitivity to etanercept
11996	0	study interventions are Bevacizumab . recurrent rectal cancer diagnosis and absolute neutrophil count greater_than equal_than one thousand, five hundred ul
11997	1	study interventions are Antibodies, Monoclonal . recurrent lymphoblastic lymphoma diagnosis and and intrathecal intraventricular therapy
11998	0	study interventions are Vorinostat . colorectal cancer diagnosis and patients must have received at least one prior chemotherapy regimen for advanced disease
11999	0	study interventions are Freund's Adjuvant . ovarian cancer diagnosis and more than four weeks since prior participation in any other investigational study

12000 rows × 2 columns

Creamos el pipeline para facilitar el uso del modelo en produccion.

```
pre = [('preproc', FunctionTransformer(preprocessor))]
```

Ahora vamos a preprocesar el texto partiendolo en tokens y lematizandolo. Despues se utilizar un modelo de bag of words y finalmente tf-idf para identificar las palabras importantes. Aprovechamos el corpus de nltk para quitar palabras conectoras que generen ruido.

```
porter = PorterStemmer()
stop = stopwords.words('english')
def tokenizer_porter(sentence):
    tokens = sentence.split()
    stemmed_tokens = [porter.stem(token) for token in tokens if token not in stop]
    return ' '.join(stemmed_tokens)

def transformer_tokenizer(df):
    df = df.apply(tokenizer_porter)
    return df

pre += [('porter', FunctionTransformer(transformer_tokenizer))]
```

```
df_eleg
```

	label	study_and_condition
0	0	study interventions are Saracatinib . recurrent verrucous carcinoma of the larynx diagnosis and patients must agree to use adequate birth control for the duration of study participation and for at least eight weeks after discontinuation of study drug
1	1	study interventions are Stem cell transplantation . hodgkin lymphoma diagnosis and history of congenital hematologic immunologic or metabolic disorder which in the estimation of the pi poses prohibitive risk to the recipient

	label	study_and_condition
2	0	study interventions are Lenograstim . recurrent adult diffuse mixed cell lymphoma diagnosis and creatinine clearance crcl greater than fifty ml per minute all tests must be performed within twenty-eight days prior to registration
3	0	study interventions are Doxorubicin . stage iii diffuse large cell lymphoma diagnosis and stages ii bulky disease defined as mass size of more than ten cm stage iii or iv ann_arbor staging patients with stage and stage ii non bulky disease are excluded from this study
4	1	study interventions are Poly I-C . prostate cancer diagnosis and unresolved iraes following prior biological therapy except that stable and managed iraes may be acceptable hypothyroidism or hypopituitarism on appropriate replacement
...	...	...
11995	0	study interventions are Prednisolone hemisuccinate . recurrent childhood large cell lymphoma diagnosis and no known hypersensitivity to etanercept
11996	0	study interventions are Bevacizumab . recurrent rectal cancer diagnosis and absolute neutrophil count greater_than equal_than one thousand, five hundred ul
11997	1	study interventions are Antibodies, Monoclonal . recurrent lymphoblastic lymphoma diagnosis and and intrathecal intraventricular therapy
11998	0	study interventions are Vorinostat . colorectal cancer diagnosis and patients must have received at least one prior chemotherapy regimen for advanced disease
11999	0	study interventions are Freund's Adjuvant . ovarian cancer diagnosis and more than four weeks since prior participation in any other investigational study

12000 rows × 2 columns

Ahora utilizamos el modelo Bag of Worda para traducir el texto a un vector numerico que representa las palabras en el mismo. Como la frecuencia de las palabras no importantes tiende a ser elevado entonces utilizamos tfidf para corregirlo.

```
tfidf = TfidfVectorizer()
```

2. Entrenamiento del modelo de clasificación de Naive Bayes

Para los parametros del modelo tomamos las dos penalidades posibles, y un rango de coeficientes de regularización para que el GridSearch pueda buscar el modelo mas optimo. También se incluye en el grid si se usa o no el IDF para ver esto como varia con el comportamiento frecuencial de los tokens. Se prueban con dos solvers.

```
Y = df_eleg['label'].astype('int')
X = df_eleg['study_and_condition']


X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.15, random_state = 98572398)
model = [('CNB',MultinomialNB())]

param_grid = {
    'tfidf__ngram_range': [(1,1)],
    'tfidf__use_idf':[False, True],
    'CNB__alpha': np.linspace(0.5, 1.5, 6),
    'CNB__fit_prior': [True, False]
}
p1 = Pipeline(pre+model)

gs = GridSearchCV(p1, param_grid, scoring='accuracy', cv=5, verbose=1, n_jobs=-1)

gs.fit(X_train, Y_train)
clf = gs.best_estimator_
print(gs.best_params_)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
{'CNB__alpha': 1.1, 'CNB__fit_prior': True, 'tfidf__ngram_range': (1, 1), 'tfidf__use_idf': True}
```

```
print("Coeficiente de determinación R^2: ",clf.score(X,Y))
```

```
Coeficiente de determinación R^2:  0.8205
```

```
y_true = Y
y_predicted = clf.predict(X)

# Note que hay que sacarle la raiz al valor
print("Root-Mean-Square Error (RMSE):",np.sqrt(mse(y_true, y_predicted)))
```

```
Root-Mean-Square Error (RMSE): 0.4236744032862972
```

El RMSE esta por debajo de 1, una muy buen metrica. Con estas dos metricas podemos ver que el modelo es correcto y se aproxima a las predicciones esperadas. Desde un punto de vista tecnico se recomienda usarlo para el negocio.

```
print('CV Accuracy: %.3f' % gs.best_score_)

print('Test Accuracy: %.3f' % clf.score(X_test, Y_test))
```

```
CV Accuracy: 0.780
Test Accuracy: 0.787
```

Finalmente tenemos metricas para poder compararlo con los otros algoritmos de clasificacion ya que estos no tienen R2 ni RMSE. Con metricas cercanas al 80% tieneun accuracy bueno y utilizable para el negocio.

```
df_export = df_eleg.copy()['study_and_condition']
df_export = preprocessor(df=df_export)
df_export = transformer_tokenizer(df_export)
tfidf_expo = TfidfVectorizer(strip_accents=None, lowercase=False, preprocessor=None, ngram_range= (1, 1),use_idf= True)
df_export =tfidf_expo.fit_transform(df_export)
```

```
df1 = pd.DataFrame(df_export.toarray(), columns=tfidf_expo.get_feature_names())
print(df1)
```

```
       0pd  0three_two9  0two_two009  101  15zero  1five  1four  1six  1six0  \
0      0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
1      0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
2      0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
3      0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
4      0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
...    ...          ...          ...  ...     ...    ...    ...   ...    ...
11995  0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
11996  0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
11997  0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
11998  0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0
11999  0.0          0.0          0.0  0.0     0.0    0.0    0.0   0.0    0.0

       1three  1two  1two0  ...  zolmitriptan  zometa  zone  zubrod   µl  \
0         0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
1         0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
2         0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
3         0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
4         0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
...       ...   ...    ...  ...           ...     ...   ...     ...  ...
11995     0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
11996     0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
11997     0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
11998     0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0
11999     0.0   0.0    0.0  ...           0.0     0.0   0.0     0.0  0.0

       µmol   ºc   ºf  ½_teaspoon   µl  µmol   υl
0       0.0  0.0  0.0         0.0  0.0   0.0  0.0
1       0.0  0.0  0.0         0.0  0.0   0.0  0.0
2       0.0  0.0  0.0         0.0  0.0   0.0  0.0
3       0.0  0.0  0.0         0.0  0.0   0.0  0.0
4       0.0  0.0  0.0         0.0  0.0   0.0  0.0
...     ...  ...  ...         ...  ...   ...  ...
11995   0.0  0.0  0.0         0.0  0.0   0.0  0.0
11996   0.0  0.0  0.0         0.0  0.0   0.0  0.0
11997   0.0  0.0  0.0         0.0  0.0   0.0  0.0
11998   0.0  0.0  0.0         0.0  0.0   0.0  0.0
11999   0.0  0.0  0.0         0.0  0.0   0.0  0.0

[12000 rows x 5509 columns]
```

```
df1.insert(loc=0, column='%label%', value=Y)
df1.to_csv('datos_finales.csv')
```