# SCRIPT DF CALIDAD

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

ruta\_proyecto = 'C:/Users/Usuario/miniconda3/SAI Dia FBS'

nombre\_fichero\_datos = 'trabajo.csv'

ruta\_completa = ruta\_proyecto + '/02\_Datos/03\_Trabajo/' + nombre\_fichero\_datos

df = pd.read\_csv(ruta\_completa)

df.rename(columns = {'CABECERA':'categoria', 'ID':'id', 'LINK':'link', 'CNO':'cno', 'TITULO OFERTA':'titulo\_oferta', 'DETALLE OFERTA':'detalle\_oferta','FIN OFERTA':'fin\_oferta','EMPRESA OFERTA':'empresa\_oferta' }, inplace=True)

df.drop(columns = {'Unnamed: 0','Unnamed: 0.1', 'link','fin\_oferta' }, inplace = True)

df.drop\_duplicates(inplace = True)

df = df.dropna(subset = ['detalle\_oferta'])

df = df[df.cno != 'CNO no informado']

# SCRIPT DF TRANSFORMADO

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import os

import re

import seaborn as sns

sns.set\_style('darkgrid')

import warnings

warnings.filterwarnings('ignore')

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfVectorizer

ruta\_proyecto = 'C:/Users/Usuario/miniconda3/SAI Dia FBS'

nombre\_fichero\_datos = 'trabajo.csv'

ruta\_completa = ruta\_proyecto + '/02\_Datos/03\_Trabajo/' + nombre\_fichero\_datos

df = pd.read\_csv(ruta\_completa)

df.rename(columns = {'CABECERA':'categoria', 'ID':'id', 'LINK':'link', 'CNO':'cno', 'TITULO OFERTA':'titulo\_oferta', 'DETALLE OFERTA':'detalle\_oferta','FIN OFERTA':'fin\_oferta','EMPRESA OFERTA':'empresa\_oferta' }, inplace=True)

df.drop(columns = {'Unnamed: 0','Unnamed: 0.1', 'link','fin\_oferta' }, inplace = True)

df.drop\_duplicates(inplace = True)

df = df.dropna(subset = ['detalle\_oferta'])

df = df[df.cno != 'CNO no informado']

CNO\_elegidos = ['3811', '3820', '2461', '5120'] #'2713',

df = df[df['cno'].isin(CNO\_elegidos)].reset\_index()

provincias\_lower = []

provincias=["Álava", "Albacete", "Alicante", "Almería", "Asturias", "Ávila", "Badajoz", "Barcelona", "Burgos", "Cáceres", "Cádiz", "Cantabria", "Castellón", "Ciudad Real", "Córdoba", "Cuenca", "Gerona", "Granada", "Guadalajara", "Guipúzcoa", "Huelva", "Huesca", "Islas Baleares", "Jaén", "La Coruña", "La Rioja", "Las Palmas", "León", "Lérida", "Lugo", "Madrid", "Málaga", "Murcia", "Navarra", "Orense", "Palencia", "Pontevedra", "Salamanca", "Santa Cruz de Tenerife", "Segovia", "Sevilla", "Soria", "Tarragona", "Teruel", "Toledo", "Valencia", "Valladolid", "Vizcaya", "Zamora", "Zaragoza"]

for i in provincias:

x = i.lower()

provincias\_lower.append(x)

stop\_words\_spanish = ['de',

'la',

'que',

'el',

'en',

'y',

'a',

'los',

'del',

'se',

'las',

'por',

'un',

'para',

'con',

'no',

'una',

'su',

'al',

'lo',

'como',

'más',

'pero',

'sus',

'le',

'ya',

'o',

'este',

'sí',

'porque',

'esta',

'entre',

'cuando',

'muy',

'sin',

'sobre',

'también',

'me',

'hasta',

'hay',

'donde',

'quien',

'desde',

'todo',

'nos',

'durante',

'todos',

'uno',

'les',

'ni',

'contra',

'otros',

'ese',

'eso',

'ante',

'ellos',

'e',

'esto',

'mí',

'antes',

'algunos',

'qué',

'unos',

'yo',

'otro',

'otras',

'otra',

'él',

'tanto',

'esa',

'estos',

'mucho',

'quienes',

'nada',

'muchos',

'cual',

'poco',

'ella',

'estar',

'estas',

'algunas',

'algo',

'nosotros',

'mi',

'mis',

'tú',

'te',

'ti',

'tu',

'tus',

'ellas',

'nosotras',

'vosostros',

'vosostras',

'os',

'mío',

'mía',

'míos',

'mías',

'tuyo',

'tuya',

'tuyos',

'tuyas',

'suyo',

'suya',

'suyos',

'suyas',

'nuestro',

'nuestra',

'nuestros',

'nuestras',

'vuestro',

'vuestra',

'vuestros',

'vuestras',

'esos',

'esas',

'estoy',

'estás',

'está',

'estamos',

'estáis',

'están',

'esté',

'estés',

'estemos',

'estéis',

'estén',

'estaré',

'estarás',

'estará',

'estaremos',

'estaréis',

'estarán',

'estaría',

'estarías',

'estaríamos',

'estaríais',

'estarían',

'estaba',

'estabas',

'estábamos',

'estabais',

'estaban',

'estuve',

'estuviste',

'estuvo',

'estuvimos',

'estuvisteis',

'estuvieron',

'estuviera',

'estuvieras',

'estuviéramos',

'estuvierais',

'estuvieran',

'estuviese',

'estuvieses',

'estuviésemos',

'estuvieseis',

'estuviesen',

'estando',

'estado',

'estada',

'estados',

'estadas',

'estad',

'he',

'has',

'ha',

'hemos',

'habéis',

'han',

'haya',

'hayas',

'hayamos',

'hayáis',

'hayan',

'habré',

'habrás',

'habrá',

'habremos',

'habréis',

'habrán',

'habría',

'habrías',

'habríamos',

'habríais',

'habrían',

'había',

'habías',

'habíamos',

'habíais',

'habían',

'hube',

'hubiste',

'hubo',

'hubimos',

'hubisteis',

'hubieron',

'hubiera',

'hubieras',

'hubiéramos',

'hubierais',

'hubieran',

'hubiese',

'hubieses',

'hubiésemos',

'hubieseis',

'hubiesen',

'habiendo',

'habido',

'habida',

'habidos',

'habidas',

'soy',

'eres',

'es',

'somos',

'sois',

'son',

'sea',

'seas',

'seamos',

'seáis',

'sean',

'seré',

'serás',

'será',

'seremos',

'seréis',

'serán',

'sería',

'serías',

'seríamos',

'seríais',

'serían',

'era',

'eras',

'éramos',

'erais',

'eran',

'fui',

'fuiste',

'fue',

'fuimos',

'fuisteis',

'fueron',

'fuera',

'fueras',

'fuéramos',

'fuerais',

'fueran',

'fuese',

'fueses',

'fuésemos',

'fueseis',

'fuesen',

'sintiendo',

'sentido',

'sentida',

'sentidos',

'sentidas',

'siente',

'sentid',

'tengo',

'tienes',

'tiene',

'tenemos',

'tenéis',

'tienen',

'tenga',

'tengas',

'tengamos',

'tengáis',

'tengan',

'tendré',

'tendrás',

'tendrá',

'tendremos',

'tendréis',

'tendrán',

'tendría',

'tendrías',

'tendríamos',

'tendríais',

'tendrían',

'tenía',

'tenías',

'teníamos',

'teníais',

'tenían',

'tuve',

'tuviste',

'tuvo',

'tuvimos',

'tuvisteis',

'tuvieron',

'tuviera',

'tuvieras',

'tuviéramos',

'tuvierais',

'tuvieran',

'tuviese',

'tuvieses',

'tuviésemos',

'tuvieseis',

'tuviesen',

'teniendo',

'tenido',

'tenida',

'tenidos',

'tenidas',

'tened']

# Creamos la función de limpoieza de textos

def clean(text):

# Quitar las expresiones entre corchetes

text = re.sub(r'\[.+?\]', ' ', str(text))

# Quita las aclaraciones entre paréntesis

text = re.sub(r'\(.+?\)', ' ', text)

# Quitar acentos

# text = unidecode.unidecode(text)

# Quitar aquello que no sean palabras o cosa que se le parezca

text = re.sub(r'\W', ' ', text)

# Quitar espacios extra en caso de haber

text = re.sub(r'\s+', ' ', text)

# Quitar un conjunto de carácteres que son intelegibles y se repiten a lo largo de los campos

text = re.sub('\r\n', ' ', text)

# Pasa el texto a minúsulas

text = text.lower()

# Quita palabras

text = re.sub(r'se busca', ' ', text)

# Quita palabras

text = re.sub(r'se necesita', ' ', text)

# Retira stopwords

text = text.split()

text = [ word for word in text if word not in stop\_words\_spanish] #stopwords.words('spanish')]

text = ' '.join(text)

return text

# Realizamos la limpieza

df['titulo\_oferta'] = df['titulo\_oferta'].apply(lambda x: clean(x))

# Creamos un listado de todas la palabras utilizadas para la oferta en una lista

words\_list = []

for i in range(0, len(df)):

text = df['titulo\_oferta'].reset\_index().iloc[i, 1]

for word in text.split():

words\_list.append(word)

words\_list

# Creamos un diccionario de las palabras y su frecuencia.

def listaPalabrasDicFrec(listaPalabras):

frecuenciaPalab = [listaPalabras.count(p) for p in listaPalabras]

return dict(list(zip(listaPalabras,frecuenciaPalab)))

# Ordemanos el diccionario por la frecuencia de la palabra

def ordenaDicFrec(dicfrec):

aux = [(dicfrec[key], key) for key in dicfrec]

aux.sort()

aux.reverse()

return aux

def remove\_provincia(text):

text = text.split()

text = [ word for word in text if word not in provincias\_lower]

text = ' '.join(text)

return text

# Realizamos la limpieza

df['titulo\_oferta'] = df['titulo\_oferta'].apply(lambda x: remove\_provincia(x))

# Creamos un listado de todas la palabras utilizadas para la oferta en una lista

words\_list = []

for i in range(0, len(df)):

text = df['titulo\_oferta'].reset\_index().iloc[i, 1]

for word in text.split():

words\_list.append(word)

words\_list

df['detalle\_oferta'] = df['detalle\_oferta'].apply(lambda x: clean(x))

# Creamos el corpus de todas la palabras utilizadas para la oferta en una lista

words\_list = []

for i in range(0, len(df)):

text = df['detalle\_oferta'].reset\_index().iloc[i, 1]

for word in text.split():

words\_list.append(word)

df['categoria'] = df['categoria'].fillna("0")

df['categoria'] = df['categoria'].apply(lambda x: clean(x))

# Creamos el corpus de todas la palabras utilizadas para la oferta en una lista

words\_list = []

for i in range(0, len(df)):

text = df['categoria'].reset\_index().iloc[i, 1]

for word in text.split():

words\_list.append(word)

# SCRIPT MODELADO

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.multiclass import OneVsRestClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn import metrics

from sklearn.metrics.pairwise import cosine\_similarity

from sklearn.metrics import roc\_auc\_score

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import plot\_confusion\_matrix

from sklearn.metrics import classification\_report

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.metrics import plot\_precision\_recall\_curve

import os

import re

import seaborn as sns

sns.set\_style('darkgrid')

import warnings

warnings.filterwarnings('ignore')

ruta\_proyecto = 'C:/Users/Usuario/OneDrive/Documentos/Proyecto Saturdays AI/SAI Dia FBS/'

nombre = 'df\_transf.pickle'

df = pd.read\_pickle(ruta\_proyecto + '/02\_Datos/03\_Trabajo/' + nombre)

# Vectorizar los textos.

Text = df['detalle\_oferta'].values

Target = df['cno'].values

word\_vectorizer = TfidfVectorizer(

sublinear\_tf=True,

#stop\_words='english',

max\_features=1500)

word\_vectorizer.fit(Text)

WordFeatures = word\_vectorizer.transform(Text)

# Realizamos el la separación nuestro conjunto de datos en Train (80%) y Test(20%)

train\_x, test\_x, train\_y, test\_y = train\_test\_split(WordFeatures, Target, random\_state=24, test\_size=0.2)

model\_GBM = OneVsRestClassifier(GradientBoostingClassifier())

model\_GBM.fit(train\_x, train\_y)

# Realizamos la prediccón para los datos de prueba

pred\_y\_GBM = model\_GBM.predict(test\_x)

# Utilizamos la función de limpoieza de textos

stop\_words\_spanish = ['de',

'la',

'que',

'el',

'en',

'y',

'a',

'los',

'del',

'se',

'las',

'por',

'un',

'para',

'con',

'no',

'una',

'su',

'al',

'lo',

'como',

'más',

'pero',

'sus',

'le',

'ya',

'o',

'este',

'sí',

'porque',

'esta',

'entre',

'cuando',

'muy',

'sin',

'sobre',

'también',

'me',

'hasta',

'hay',

'donde',

'quien',

'desde',

'todo',

'nos',

'durante',

'todos',

'uno',

'les',

'ni',

'contra',

'otros',

'ese',

'eso',

'ante',

'ellos',

'e',

'esto',

'mí',

'antes',

'algunos',

'qué',

'unos',

'yo',

'otro',

'otras',

'otra',

'él',

'tanto',

'esa',

'estos',

'mucho',

'quienes',

'nada',

'muchos',

'cual',

'poco',

'ella',

'estar',

'estas',

'algunas',

'algo',

'nosotros',

'mi',

'mis',

'tú',

'te',

'ti',

'tu',

'tus',

'ellas',

'nosotras',

'vosostros',

'vosostras',

'os',

'mío',

'mía',

'míos',

'mías',

'tuyo',

'tuya',

'tuyos',

'tuyas',

'suyo',

'suya',

'suyos',

'suyas',

'nuestro',

'nuestra',

'nuestros',

'nuestras',

'vuestro',

'vuestra',

'vuestros',

'vuestras',

'esos',

'esas',

'estoy',

'estás',

'está',

'estamos',

'estáis',

'están',

'esté',

'estés',

'estemos',

'estéis',

'estén',

'estaré',

'estarás',

'estará',

'estaremos',

'estaréis',

'estarán',

'estaría',

'estarías',

'estaríamos',

'estaríais',

'estarían',

'estaba',

'estabas',

'estábamos',

'estabais',

'estaban',

'estuve',

'estuviste',

'estuvo',

'estuvimos',

'estuvisteis',

'estuvieron',

'estuviera',

'estuvieras',

'estuviéramos',

'estuvierais',

'estuvieran',

'estuviese',

'estuvieses',

'estuviésemos',

'estuvieseis',

'estuviesen',

'estando',

'estado',

'estada',

'estados',

'estadas',

'estad',

'he',

'has',

'ha',

'hemos',

'habéis',

'han',

'haya',

'hayas',

'hayamos',

'hayáis',

'hayan',

'habré',

'habrás',

'habrá',

'habremos',

'habréis',

'habrán',

'habría',

'habrías',

'habríamos',

'habríais',

'habrían',

'había',

'habías',

'habíamos',

'habíais',

'habían',

'hube',

'hubiste',

'hubo',

'hubimos',

'hubisteis',

'hubieron',

'hubiera',

'hubieras',

'hubiéramos',

'hubierais',

'hubieran',

'hubiese',

'hubieses',

'hubiésemos',

'hubieseis',

'hubiesen',

'habiendo',

'habido',

'habida',

'habidos',

'habidas',

'soy',

'eres',

'es',

'somos',

'sois',

'son',

'sea',

'seas',

'seamos',

'seáis',

'sean',

'seré',

'serás',

'será',

'seremos',

'seréis',

'serán',

'sería',

'serías',

'seríamos',

'seríais',

'serían',

'era',

'eras',

'éramos',

'erais',

'eran',

'fui',

'fuiste',

'fue',

'fuimos',

'fuisteis',

'fueron',

'fuera',

'fueras',

'fuéramos',

'fuerais',

'fueran',

'fuese',

'fueses',

'fuésemos',

'fueseis',

'fuesen',

'sintiendo',

'sentido',

'sentida',

'sentidos',

'sentidas',

'siente',

'sentid',

'tengo',

'tienes',

'tiene',

'tenemos',

'tenéis',

'tienen',

'tenga',

'tengas',

'tengamos',

'tengáis',

'tengan',

'tendré',

'tendrás',

'tendrá',

'tendremos',

'tendréis',

'tendrán',

'tendría',

'tendrías',

'tendríamos',

'tendríais',

'tendrían',

'tenía',

'tenías',

'teníamos',

'teníais',

'tenían',

'tuve',

'tuviste',

'tuvo',

'tuvimos',

'tuvisteis',

'tuvieron',

'tuviera',

'tuvieras',

'tuviéramos',

'tuvierais',

'tuvieran',

'tuviese',

'tuvieses',

'tuviésemos',

'tuvieseis',

'tuviesen',

'teniendo',

'tenido',

'tenida',

'tenidos',

'tenidas',

'tened']

def clean(text):

# Quitar las expresiones entre corchetes

text = re.sub(r'\[.+?\]', ' ', str(text))

# Quita las aclaraciones entre paréntesis

text = re.sub(r'\(.+?\)', ' ', text)

# Quitar acentos

# text = unidecode.unidecode(text)

# Quitar aquello que no sean palabras o cosa que se le parezca

text = re.sub(r'\W', ' ', text)

# Quitar espacios extra en caso de haber

text = re.sub(r'\s+', ' ', text)

# Quitar un conjunto de carácteres que son intelegibles y se repiten a lo largo de los campos

text = re.sub('\r\n', ' ', text)

# Pasa el texto a minúsulas

text = text.lower()

# Quita palabras

text = re.sub(r'se busca', ' ', text)

# Quita palabras

text = re.sub(r'se necesita', ' ', text)

# Retira stopwords

text = text.split()

text = [ word for word in text if word not in stop\_words\_spanish ]# stopwords.words('spanish')]

text = ' '.join(text)

return text

text = """Sentar a los clientes en sus mesas Tomar nota de sus pedidos Llevar la comida y bebida a las mesas Asegurarse de que los clientes están a gusto y que disponen de todo lo que necesitan Recibir el pago al final de la comida"""

text = [clean(text)]

WordF = word\_vectorizer.transform(text)

model\_GBM.predict(WordF)