Arbol de decisiones 15_07_23

July 15, 2023

[1]:
```
#Vamos a tratar de predecir si una canción llega al top 1 de billboard o no.
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
import pydot
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from IPython.display import Image as PImage
from subprocess import check_call
from PIL import Image, ImageDraw, ImageFont
```

[2]:
```
artists_billboard = pd.read_csv("artists_billboard_fix3.csv")
artists_billboard.head()
```

[2]:
```
   id                 title  \
0   0  Small Town Throwdown
1   1             Bang Bang
2   2                Timber
3   3       Sweater Weather
4   4             Automatic


                                              artist        mood  \
0  BRANTLEY GILBERT featuring JUSTIN MOORE & THOM…  Brooding
1              JESSIE J, ARIANA GRANDE & NICKI MINAJ  Energizing
2                            PITBULL featuring KE$HA     Excited
3                                  THE NEIGHBOURHOOD    Brooding
4                                    MIRANDA LAMBERT    Yearning

          tempo               genre artist_type  chart_date  durationSeg  top  \
0  Medium Tempo         Traditional        Male    20140628        191.0    0
1  Medium Tempo                 Pop      Female    20140816        368.0    0
2  Medium Tempo               Urban       Mixed    20140118        223.0    1
```

```
3  Medium Tempo  Alternative & Punk        Male    20140104         206.0    0
4  Medium Tempo         Traditional      Female    20140301         232.0    0

   anioNacimiento
0          1975.0
1          1989.0
2          1993.0
3          1989.0
4             0.0
```

[3]:
```
#Cuantos registros tengo de cada clase
artists_billboard.groupby('top').size()
```

[3]:
```
top
0    494
1    141
dtype: int64
```

[4]:
```
artists_billboard.describe()
```

[4]:
```
               id    chart_date  durationSeg         top  anioNacimiento
count  635.000000  6.350000e+02   635.000000  635.000000      635.000000
mean   317.000000  2.013036e+07   321.768504    0.222047     1548.590551
std    183.452991  2.617996e+04   633.753787    0.415950      820.470454
min      0.000000  2.004021e+07     0.000000    0.000000        0.000000
25%    158.500000  2.014010e+07   200.000000    0.000000     1969.000000
50%    317.000000  2.014051e+07   232.000000    0.000000     1981.000000
75%    475.500000  2.014101e+07   266.500000    0.000000     1986.000000
max    634.000000  2.015031e+07  6840.000000    1.000000     1999.000000
```

[5]:
```
#Cambiar los 0 por el valor None
def edad_fix(anio):
    if anio==0:
        return None
    return anio
```

[6]:
```
#Reemplazo de 0's por el valor "None"
artists_billboard['anioNacimiento']=artists_billboard.apply(lambda x:␣
 ↪edad_fix(x['anioNacimiento']), axis=1)
```

[7]:
```
artists_billboard.head()
```

[7]:
```
   id                 title  \
0   0  Small Town Throwdown
1   1             Bang Bang
2   2                Timber
3   3       Sweater Weather
```

```
4   4             Automatic

                                           artist        mood  \
0  BRANTLEY GILBERT featuring JUSTIN MOORE & THOM…    Brooding
1               JESSIE J, ARIANA GRANDE & NICKI MINAJ  Energizing
2                             PITBULL featuring KE$HA     Excited
3                                   THE NEIGHBOURHOOD    Brooding
4                                     MIRANDA LAMBERT    Yearning

          tempo               genre artist_type  chart_date  durationSeg  top  \
0  Medium Tempo         Traditional        Male    20140628        191.0    0
1  Medium Tempo                 Pop      Female    20140816        368.0    0
2  Medium Tempo               Urban       Mixed    20140118        223.0    1
3  Medium Tempo  Alternative & Punk        Male    20140104        206.0    0
4  Medium Tempo         Traditional      Female    20140301        232.0    0

   anioNacimiento
0          1975.0
1          1989.0
2          1993.0
3          1989.0
4             NaN
```

[8]:
```python
artists_billboard.describe()
```

[8]:
```
               id    chart_date  durationSeg         top  anioNacimiento
count  635.000000  6.350000e+02   635.000000  635.000000      496.000000
mean   317.000000  2.013036e+07   321.768504    0.222047     1982.570565
std    183.452991  2.617996e+04   633.753787    0.415950        8.346478
min      0.000000  2.004021e+07     0.000000    0.000000     1919.000000
25%    158.500000  2.014010e+07   200.000000    0.000000     1978.000000
50%    317.000000  2.014051e+07   232.000000    0.000000     1984.000000
75%    475.500000  2.014101e+07   266.500000    0.000000     1988.000000
max    634.000000  2.015031e+07  6840.000000    1.000000     1999.000000
```

[10]:
```python
#Función para calcular las edades en las que estuvieron en el billboard
def calcula_edad(anio,cuando):
    cad = str(cuando)
    momento = cad[:4]
    if anio==0.0:
        return None
    return int(momento) - anio
```

[11]:
```python
artists_billboard['edad_en_billboard']=artists_billboard.apply(lambda x:␣
 ↪calcula_edad(x['anioNacimiento'],x['chart_date']), axis=1)
```

[12]:
```python
artists_billboard.head()
```

[12]:

```
   id                 title  \
0   0  Small Town Throwdown   
1   1             Bang Bang   
2   2                Timber   
3   3       Sweater Weather   
4   4             Automatic   

                                              artist        mood  \
0  BRANTLEY GILBERT featuring JUSTIN MOORE & THOM…    Brooding   
1               JESSIE J, ARIANA GRANDE & NICKI MINAJ  Energizing   
2                             PITBULL featuring KE$HA     Excited   
3                                   THE NEIGHBOURHOOD    Brooding   
4                                     MIRANDA LAMBERT    Yearning   

          tempo               genre artist_type  chart_date  durationSeg  top  \
0  Medium Tempo         Traditional        Male    20140628        191.0    0   
1  Medium Tempo                 Pop      Female    20140816        368.0    0   
2  Medium Tempo               Urban       Mixed    20140118        223.0    1   
3  Medium Tempo  Alternative & Punk        Male    20140104        206.0    0   
4  Medium Tempo         Traditional      Female    20140301        232.0    0   

   anioNacimiento  edad_en_billboard  
0          1975.0               39.0  
1          1989.0               25.0  
2          1993.0               21.0  
3          1989.0               25.0  
4             NaN                NaN  
```

[13]:

```python
#Asignar valores al azar en el rango de media-std a media+std que es de 21 a 37
age_avg = artists_billboard['edad_en_billboard'].mean()
age_std = artists_billboard['edad_en_billboard'].std()
age_null_count = artists_billboard['edad_en_billboard'].isnull().sum()
age_null_random_list = np.random.randint(age_avg - age_std, age_avg + age_std, size=age_null_count)
```

[14]:

```python
age_null_random_list[:5]
```

[14]:

```
array([35, 36, 26, 33, 21])
```

[15]:

```python
conValoresNulos = np.isnan(artists_billboard['edad_en_billboard'])
conValoresNulos
```

[15]:

```
0    False
1    False
2    False
3    False
4     True
```

```
      …
630    False
631    False
632    False
633    False
634    False
Name: edad_en_billboard, Length: 635, dtype: bool
```

[16]:

```
artists_billboard.loc[np.isnan(artists_billboard['edad_en_billboard']),␣
 ↪'edad_en_billboard'] = age_null_random_list
artists_billboard['edad_en_billboard'] = artists_billboard['edad_en_billboard'].
 ↪astype(int)
artists_billboard.head()
```

[16]:

```
   id                 title  \
0   0  Small Town Throwdown
1   1             Bang Bang
2   2                Timber
3   3       Sweater Weather
4   4             Automatic

                                              artist        mood  \
0  BRANTLEY GILBERT featuring JUSTIN MOORE & THOM…    Brooding
1               JESSIE J, ARIANA GRANDE & NICKI MINAJ  Energizing
2                             PITBULL featuring KE$HA     Excited
3                                 THE NEIGHBOURHOOD    Brooding
4                                   MIRANDA LAMBERT    Yearning

          tempo                genre artist_type  chart_date  durationSeg  top  \
0  Medium Tempo          Traditional        Male    20140628        191.0    0
1  Medium Tempo                  Pop      Female    20140816        368.0    0
2  Medium Tempo                Urban       Mixed    20140118        223.0    1
3  Medium Tempo  Alternative & Punk        Male    20140104        206.0    0
4  Medium Tempo          Traditional      Female    20140301        232.0    0

   anioNacimiento  edad_en_billboard
0          1975.0                 39
1          1989.0                 25
2          1993.0                 21
3          1989.0                 25
4             NaN                 35
```

[17]:

```
print("Edad Promedio: " + str(age_avg))
print("Desvió Std Edad: " + str(age_std))
print("Intervalo para asignar edad aleatoria: " + str(int(age_avg - age_std)) +␣
 ↪" a " + str(int(age_avg + age_std)))
```

```
Edad Promedio: 30.10282258064516
Desvió Std Edad: 8.40078832861513
Intervalo para asignar edad aleatoria: 21 a 38
```

[18]:
```
#Voy a empezara mappear el mood para que sean variables categóricas
artists_billboard['moodEncoded'] = artists_billboard['mood'].map(
    {
        'Energizing': 6,
        'Empowering': 6,
        'Cool': 5,
        'Yearning': 4,
        'Excited': 5,
        'Defiant': 3,
        'Sensual': 2,
        'Gritty': 3,
        'Sophisticated': 4,
        'Aggressive': 4,
        'Fiery': 4,
        'Urgent': 3,
        'Rowdy': 4,
        'Sentimental': 4,
        'Easygoing': 1,
        'Melancholy': 4,
        'Romantic': 2,
        'Peaceful': 1,
        'Brooding': 4,
        'Upbeat': 5,
        'Stirring': 5,
        'Lively': 5,
        'Other': 0,
        '':0
    }).astype(int)
```

[19]:
```
#Mapeo del tempo
artists_billboard['tempoEncoded'] = artists_billboard['tempo'].map( {'Fast␣
 ↪Tempo': 0, 'Medium Tempo': 2, 'Slow Tempo': 1, '': 0} ).astype(int)
```

[36]:
```
#Mapeo del genero
artists_billboard['genreEncoded'] = artists_billboard['genre'].map( {'Urban': 4,
'Pop': 3,
'Traditional': 2, 'Alternative & Punk': 1,
'Electronica': 1, 'Rock': 1, 'Soundtrack': 0, 'Jazz': 0, 'Other':0,'':0}
).astype(int)
```

[37]:
```
#Mapeo del tipo de artista
artists_billboard['artist_typeEncoded'] = artists_billboard['artist_type'].map(␣
 ↪{'Female': 2, 'Male': 3, 'Mixed': 1, '': 0} ).astype(int)
```

[38]:

```
#Mapeo de la edad
artists_billboard.loc[ artists_billboard['edad_en_billboard'] <= 21,␣
 ↪'edadEncoded'] = 0
artists_billboard.loc[(artists_billboard['edad_en_billboard'] > 21) &␣
 ↪(artists_billboard['edad_en_billboard'] <= 26), 'edadEncoded'] = 1
artists_billboard.loc[(artists_billboard['edad_en_billboard'] > 26) &␣
 ↪(artists_billboard['edad_en_billboard'] <= 30), 'edadEncoded'] = 2
artists_billboard.loc[(artists_billboard['edad_en_billboard'] > 30) &␣
 ↪(artists_billboard['edad_en_billboard'] <= 40), 'edadEncoded'] = 3
artists_billboard.loc[ artists_billboard['edad_en_billboard'] > 40,␣
 ↪'edadEncoded'] = 4
```

[39]:

```
#Mapeo de duracion en segundos
artists_billboard.loc[ artists_billboard['durationSeg'] <= 150,␣
 ↪'durationEncoded'] = 0
artists_billboard.loc[(artists_billboard['durationSeg'] > 150) &␣
 ↪(artists_billboard['durationSeg'] <= 180), 'durationEncoded'] = 1
artists_billboard.loc[(artists_billboard['durationSeg'] > 180) &␣
 ↪(artists_billboard['durationSeg'] <= 210), 'durationEncoded'] = 2
artists_billboard.loc[(artists_billboard['durationSeg'] > 210) &␣
 ↪(artists_billboard['durationSeg'] <= 240), 'durationEncoded'] = 3
artists_billboard.loc[(artists_billboard['durationSeg'] > 240) &␣
 ↪(artists_billboard['durationSeg'] <= 270), 'durationEncoded'] = 4
artists_billboard.loc[(artists_billboard['durationSeg'] > 270) &␣
 ↪(artists_billboard['durationSeg'] <= 300), 'durationEncoded'] = 5
artists_billboard.loc[ artists_billboard['durationSeg'] > 300,␣
 ↪'durationEncoded'] = 6
```

[40]:

```
artists_billboard.head()
```

[40]:

```
   id                 title  \
0   0  Small Town Throwdown
1   1             Bang Bang
2   2                Timber
3   3       Sweater Weather
4   4             Automatic

                                              artist        mood  \
0  BRANTLEY GILBERT featuring JUSTIN MOORE & THOM…    Brooding
1              JESSIE J, ARIANA GRANDE & NICKI MINAJ  Energizing
2                            PITBULL featuring KE$HA     Excited
3                                  THE NEIGHBOURHOOD    Brooding
4                                    MIRANDA LAMBERT    Yearning

          tempo               genre artist_type  chart_date  durationSeg  top  \
0  Medium Tempo         Traditional        Male    20140628        191.0    0
```

```
1  Medium Tempo                  Pop      Female    20140816        368.0    0
2  Medium Tempo                Urban       Mixed    20140118        223.0    1
3  Medium Tempo  Alternative & Punk         Male    20140104        206.0    0
4  Medium Tempo          Traditional      Female    20140301        232.0    0

   anioNacimiento  edad_en_billboard  moodEncoded  tempoEncoded  \
0          1975.0                 39            4             2
1          1989.0                 25            6             2
2          1993.0                 21            5             2
3          1989.0                 25            4             2
4             NaN                 35            4             2

   artist_typeEncoded  edadEncoded  durationEncoded  genreEncoded
0                   3          3.0              2.0             2
1                   2          1.0              6.0             3
2                   1          0.0              3.0             4
3                   3          1.0              2.0             1
4                   2          3.0              3.0             2
```

[41]:
```
#Tirar las columnas que no necesito
drop_elements =␣
 ↪['id','title','artist','mood','tempo','genre','artist_type','chart_date','anioNacimiento','
artists_encoded = artists_billboard.drop(columns = drop_elements)
```

[58]:
```
artists_encoded.head()
```

[58]:
```
   top  moodEncoded  tempoEncoded  artist_typeEncoded  edadEncoded  \
0    0            4             2                   3          3.0
1    0            6             2                   2          1.0
2    1            5             2                   1          0.0
3    0            4             2                   3          1.0
4    0            4             2                   2          3.0

   durationEncoded  genreEncoded
0              2.0             2
1              6.0             3
2              3.0             4
3              2.0             1
4              3.0             2
```

[59]:
```
#Encontrar la profundidad máxima de mi árbol
cv = KFold(n_splits=10) # Numero deseado de "folds" que haremos
accuracies = []
max_attributes = len(list(artists_encoded))
depth_range = range(1, max_attributes + 1)
for depth in depth_range: #range(1,7)
    fold_accuracy = []
    tree_model = tree.DecisionTreeClassifier(criterion='entropy',
    min_samples_split=20, min_samples_leaf=5, max_depth = depth,␣
 ↪class_weight={1:3.5})
    for train_fold, valid_fold in cv.split(artists_encoded):
        f_train = artists_encoded.loc[train_fold]
        f_valid = artists_encoded.loc[valid_fold]
        model = tree_model.fit(X = f_train.drop(['top'], axis=1), y =␣
 ↪f_train["top"])
        valid_acc = model.score(X = f_valid.drop(['top'], axis=1), y =␣
 ↪f_valid["top"])
        # calculamos la precision con el segmento de validacion
        fold_accuracy.append(valid_acc)
    avg = sum(fold_accuracy)/len(fold_accuracy)
    accuracies.append(avg)
```

[60]:
```
df = pd.DataFrame({"Max Depth": depth_range, "Average Accuracy": accuracies})
df = df[["Max Depth", "Average Accuracy"]]
print(df.to_string(index=False))
```

```
Max Depth  Average Accuracy
        1          0.556101
        2          0.556126
        3          0.564038
        4          0.645685
        5          0.628547
        6          0.654985
        7          0.648859
```

[61]:
```
# Crear arrays de entrenamiento y las etiquetas que indican si llegó a top o no
y_train = artists_encoded['top']
X_train = artists_encoded.drop(['top'], axis=1).values
```

[62]:
```
decision_tree = tree.
 ↪DecisionTreeClassifier(criterion='entropy',min_samples_split=20,␣
 ↪min_samples_leaf=5, max_depth = 4, class_weight={1:3.5})
```

[63]:
```
decision_tree.fit(X_train, y_train)
```

[63]:
```
DecisionTreeClassifier(class_weight={1: 3.5}, criterion='entropy', max_depth=4,
                       min_samples_leaf=5, min_samples_split=20)
```

[64]:
```
with open(r"tree1.dot", 'w') as f:
    f = tree.export_graphviz(decision_tree,
                             out_file=f,
                             max_depth = 7,
                             impurity = True,
                                feature_names = list(artists_encoded.drop(['top'],␣
↪axis=1)),
                                class_names = ['No', 'N1Billboard'],
                                rounded = True,
                                filled= True )
```

[65]:

```
x_test = pd.DataFrame(columns=('top','moodEncoded', 'tempoEncoded',␣
↪'genreEncoded','artist_typeEncoded','edadEncoded','durationEncoded '))
```

[66]:

```
x_test.loc[0] = (1,5,2,4,1,0,3)
```

[67]:

```
y_pred = decision_tree.predict(x_test.drop(columns=['top']))
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-
packages/sklearn/base.py:439: UserWarning: X has feature names, but
DecisionTreeClassifier was fitted without feature names
  f"X has feature names, but {self.__class__.__name__} was fitted without"
```

[68]:

```
print("Prediccion: " + str(y_pred))
```

```
Prediccion: [1]
```

[69]:

```
y_proba = decision_tree.predict_proba(x_test.drop(columns = ['top']))
print("Probabilidad de Acierto: " + str(y_proba[0][y_pred]* 100)+"%")
```

```
Probabilidad de Acierto: [62.60162602]%
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-
packages/sklearn/base.py:439: UserWarning: X has feature names, but
DecisionTreeClassifier was fitted without feature names
  f"X has feature names, but {self.__class__.__name__} was fitted without"
```

[70]:

```
x_test = pd.DataFrame(columns=('top','moodEncoded', 'tempoEncoded',␣
↪'genreEncoded','artist_typeEncoded','edadEncoded','durationEncoded '))
```

[73]:

```
x_test.loc[0] = (0,4,2,1,3,2,3)
```

[74]:

```
y_pred = decision_tree.predict(x_test.drop(columns=['top']))
print("Prediccion: " + str(y_pred))
```

```
Prediccion: [1]
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-
packages/sklearn/base.py:439: UserWarning: X has feature names, but
DecisionTreeClassifier was fitted without feature names
  f"X has feature names, but {self.__class__.__name__} was fitted without"
```

[75]:

```
y_proba = decision_tree.predict_proba(x_test.drop(['top'], axis = 1))
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-
packages/sklearn/base.py:439: UserWarning: X has feature names, but
DecisionTreeClassifier was fitted without feature names
  f"X has feature names, but {self.__class__.__name__} was fitted without"
```

[76]:

```
print("Probabilidad de Acierto: " + str(y_proba[0][y_pred]* 100)+"%")
```

```
Probabilidad de Acierto: [54.40414508]%
```

[ ]:

```
```