In [75]:

```
%matplotlib inline
```

Uso de la librería NLTK para identificar el género

1.- Librerias

In [76]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import re

import nltk
from nltk.corpus import names
from sklearn import cross_validation
import random


import dateutil
#Hay que istalar esta librería que hace el parseo del user agent
#pip install pyyaml ua-parser user-agents



#Para pintar gráficos vistosos usamos seaborn:
import seaborn as sns

#y creamos la paleta:
sns.set_palette("deep", desat=.6)
sns.set_context(rc={"figure.figsize": (8, 4)})
```

2.- Descripcion de los datos

DESPUES DE ANONIMIZAR Y SELECCIONAR ÚNICAMENTE LAS VARIABLES QUE QUEREMOS

num_columna	Nombre	Descripción	Variable
1	ciudad	ciuda de origen del usuario	discreta
2	email_server	servidor de email del usuario	discreta
3	**edad**	edad del usuario (variable objetivo)	discreta
4	**genero**	genero del usuario (variable objetivo)	discreta
6	hora_visita	hora en que el usuario hace la visita	discreta
7	is_weekend	fin de semana	discreta
8	nombre_final	nombre del usuario	discreta
9	os	sistema operativo	discreta
10	pais	pais en el user agent	discreta
11	rango horario	momento del día en que se conecta el usuario	discreta
12	time_zone	zona horaria del usuario	discreta
13	ua_browser_family	familia del navegador en el user agent	discreta
14	ua_device	dispositivo que utiliza el usuario segun user agent	discreta
15	ua_device_family	familia del dispositivo en el user agent	discreta
16	ua_is_bot	si es un robot	discreta
17	ua_is_movile	si es un movil	discreta
19	ua_is_pc	si es un pc	discreta
20	ua_is_tablet	si es una tablet	discreta
21	ua_is_tounch_capable	si es táctil	discreta
22	ua_os_family	familia sistema operativo	discreta
23	weekday	dia de la semana	discreta
24	id_hotspots	id del local	discreta

```
-Faltaría saber si se ha conectado con facebook, google o email (deb
ería hacerlo en la recolección de variables), así como rellenar los
nulos con un valor ("vacio")
-También faltaría la categoría del local en que se ha conectado y ha
cer algo con las provincias.
```

3.- Carga de los datos

Cargamos los datos que hemos limpiado anteriormente y guardado en un csv para cargarlos más fácilmente). Al final del ejercicio habría que integrarlo todo en un único proceso para su uso.

In [77]:

```
df = pd.read_csv('../csv/datos_explorados.csv')

#borro la columna unnamed
df.drop('Unnamed: 0', axis=1,inplace=True)

#y quito ciudad, ua_os_family  y ua_device_family

df.drop(['ua_os_family','ua_device',u'ciudad','ua_is_pc'], axis=1,inplace=T
rue)


print df.columns
```

```
Index([u'email_server', u'edad', u'genero', u'hora_visita', u'idioma', u'i
s_weekend', u'nombre_final', u'os', u'pais', u'rango_horario', u'timezone',
u'ua_browser_family', u'ua_device_family', u'ua_is_bot', u'ua_is_movile',
u'ua_is_tablet', u'ua_is_tounch_capable', u'weekday', u'id_hotspots'], dtyp
e='object')
```

Nos quedamos con los datos que necesitamos, en este caso va a ser país,nombre_final y género

In [78]:

```
##Nos creamos un dataFrame que guardará el ranking de resultados:
```

In [79]:

```
df_resultados = pd.DataFrame
clasificador =[]
resultado_clasificador = []
```

In [80]:

```
df_final = df[["pais","nombre_final", "genero"]]

#Elimino los nulos
df_final.loc[:,("nombre_final")] = df_final.nombre_final.fillna("sin_nombr
e")
print df_final.head()
print df_final.genero.unique()
```

```
    pais nombre_final genero
0  vacio         Abel    NaN
1     NL         Gert    NaN
2     GB       Fergus    NaN
3     IE       Fergus    NaN
4     ES         Paco    NaN
[nan 'male' 'female' 'other']
```

In [81]:

```
def gender_features(word):
    return {'last_letter': word[-1]}

def country_features(pais):
    return{'pais': pais}

def gender_country_features(name, pais):
    return {'last_letter': name[-1],
            'pais': pais}

#Vamos a dividir entre españa y extranjero
def prepara_pais(df_final):
    df_final.pais = ["ES" if pais == 'ES' else 'NO_ES' for pais in df_fina
l.pais]
    return df_final

def elimina_duplicados(df_final):
    df_final = df_final.drop_duplicates(["nombre_final","pais","genero"])
    return df_final

df_final = prepara_pais(df_final)
df_final = elimina_duplicados(df_final)

print df_final.head()
```

```
    pais nombre_final genero
0  NO_ES         Abel    NaN
1  NO_ES         Gert    NaN
2  NO_ES       Fergus    NaN
4     ES         Paco    NaN
5  NO_ES      Dongwan    NaN
```

In [82]:

```
#importamos el corpus de nltk y añadimos el nuestro

hombres_df = df_final[df_final.genero=='male'][["nombre_final","pais"]]
mujeres_df = df_final[df_final.genero=='female'][["nombre_final","pais"]]


hombres_nltk =  names.words('male.txt')
mujeres_nltk =  names.words('female.txt')

#Me creo un DataFrame para poder compararlo con el mío
df_hombres_nltk = pd.DataFrame()
df_mujeres_nltk = pd.DataFrame()

df_hombres_nltk["nombre_final"] = hombres_nltk
df_mujeres_nltk["nombre_final"] = mujeres_nltk

#Como el corpus que tenemos de nltk es de nombres no españoles, le ponemos
el país NO_ES
df_hombres_nltk["pais"] = np.repeat('NO_ES',len(hombres_nltk))
df_mujeres_nltk["pais"] = np.repeat('NO_ES',len(mujeres_nltk))

hombres = pd.concat([hombres_df,df_hombres_nltk],ignore_index=True)
mujeres = pd.concat([mujeres_df,df_mujeres_nltk],ignore_index=True)
hombres["genero"] = np.repeat('male',len(hombres))
mujeres["genero"] = np.repeat('female', len(mujeres))

#concatenamos los dos dataframes

gender_dataframe = pd.concat([hombres,mujeres],ignore_index=True)
#Elimino las filas duplicadas que son muchas
gender_dataframe.drop_duplicates(["nombre_final","pais","genero"], inplace
= True)
#Vamos a limpiar un poco el resultado

print "Longitud del dataframe:" , len(gender_dataframe["nombre_final"])
def borra_solo_una_letra(gender_dataframe):
    gender_dataframe = gender_dataframe[gender_dataframe["nombre_final"].st
r.len()>1]
    return gender_dataframe

#Borro los nombres con solo una letra
gender_dataframe = borra_solo_una_letra(gender_dataframe)
print "Longitud del dataframe sin los nombres de una letra:" , len(gender_d
ataframe)
```

```
Longitud del dataframe: 10010
Longitud del dataframe sin los nombres de una letra: 9973
```

In [83]:

```
# Voy a crear la lista de features con su etiqueta y hacemos un suffle para
que se mezcle
# porque está male delante de female

featuresets = []

for row in gender_dataframe.iterrows():
    featuresets.append((gender_country_features(row[1].nombre_final,ro
w[1].pais)
                        ,row[1].genero))


random.shuffle(featuresets)

train_set, test_set = featuresets[2000:], featuresets[:2000]

classifier = nltk.NaiveBayesClassifier.train(train_set)
```

In [84]:

```
# vemaos lo que acierta y las letras finales mejores:
resultado_1= nltk.classify.accuracy(classifier, test_set)
print "Score del algoritmo:" , resultado_1

clasificador.append("clasifier_1")
resultado_clasificador.append(resultado_1)

classifier.show_most_informative_features(5)
```

```
Score del algoritmo: 0.7525
Most Informative Features
             last_letter = 'k'              male : female =     16.5 : 1.0
             last_letter = u'f'             male : female =     16.0 : 1.0
             last_letter = u'a'           female : male   =     14.8 : 1.0
             last_letter = u'v'             male : female =     13.1 : 1.0
             last_letter = 'c'              male : female =     11.2 : 1.0
```

- El resultado es del 75% de acierto que es mucha más que el 50% si lo hicíeramos a ojo
- El parámreto "Most Informative Features" viene a decir:

  pais = 'RO' male : female = 6.8 : 1.0

  Si el país es 'RO' 6.8 veces hombre por 1 mujer.

In [85]:

```
#Vamos a hacer una funcion cross-validation para ver si hay overfitting o n
o

def crossValidationGender(featuresets, cortes):
    training_set = featuresets
    cv = cross_validation.KFold(len(training_set), n_folds=cortes,  shuffl
e=False, random_state=None)
    accum_accuracy = 0
    max_accuracy = 0
    min_accuracy = 1

    for traincv, testcv in cv:
        classifier = nltk.NaiveBayesClassifier.train(training_set[trainc
v[0]:traincv[len(traincv)-1]])
        accuracy = nltk.classify.util.accuracy(classifier, training_set[tes
tcv[0]:testcv[len(testcv)-1]])
        accum_accuracy += accuracy
        if accuracy > max_accuracy:
            max_accuracy = accuracy
        if accuracy < min_accuracy:
            min_accuracy = accuracy

    print "Accuracy average = ", accum_accuracy/cortes
    print "Max accuracy = " , max_accuracy
    print "Min Accuracy = " , min_accuracy
```

In [86]:

```
# y la probamos:

crossValidationGender(featuresets,100)
```

```
Accuracy average =  0.754384662956
Max accuracy =  0.909090909091
Min Accuracy =  0.656565656566
```

In [87]:

```
# Vamos a definir más características:
#  - La ultima letra
#  - Las  2 últimas letras
#  - El pais
# Si tiene una letra

def gender_country_features_2(word, pais):
    return {'suffix1': word[-1:],
            'suffix2': word[-2:],
            'pais': pais}
```

In [88]:

```
featuresets2 = []

for row in gender_dataframe.iterrows():
    featuresets2.append((gender_country_features_2(row[1].nombre_final,ro
w[1].pais)
                         ,row[1].genero))

#Hago que los elementos sean únicos, porque hay muchos repetidos
random.shuffle(featuresets2)

num_train = int(round(len(featuresets2) * 0.4,0))

train_set, test_set = featuresets2[num_train:], featuresets2[:num_train]
classifier_2 = nltk.NaiveBayesClassifier.train(train_set)
resultado_2 = nltk.classify.accuracy(classifier_2, test_set)
print resultado_2
classifier_2.show_most_informative_features(5)

clasificador.append("classifier_2")
clasificador.append(resultado_2)
```

```
0.774880922537
Most Informative Features
                suffix2 = u'na'          female : male   =     60.3 : 1.0
                suffix2 = 'da'           female : male   =     41.5 : 1.0
                suffix2 = 'sa'           female : male   =     31.1 : 1.0
                suffix2 = u'rd'            male : female =     29.1 : 1.0
                suffix2 = 'ck'             male : female =     28.9 : 1.0
```

In [89]:

```
# y ahora cross-validation
crossValidationGender(featuresets2,30)
```

```
Accuracy average =  0.780242723158
Max accuracy =  0.816265060241
Min Accuracy =  0.737160120846
```

In [90]:

```
#Así sería el pipeline con los datos

adivina = df[["pais","nombre_final", "genero"]]
adivina = adivina[pd.isnull(adivina.genero)]
adivina = prepara_pais(adivina)
adivina = elimina_duplicados(adivina)

i= 0

for row in adivina.iterrows():
    i+=1
    feature = (gender_country_features_2(row[1].nombre_final,row[1].pais))
    print row[0], ":" , row[1].nombre_final , ", ",row[1].pais, ":" , class
ifier_2.classify(feature)
    if i == 10:
        break
```

```
0 : Abel ,  NO_ES : male
1 : Gert ,  NO_ES : male
2 : Fergus ,  NO_ES : male
4 : Paco ,  ES : male
5 : Dongwan ,  NO_ES : male
6 : Lauren ,  NO_ES : male
8 : Ele ,  NO_ES : female
142 : Alejandro ,  NO_ES : male
143 : dan ,  NO_ES : male
172 : linn ,  NO_ES : female
```

In [91]:

```
for row in gender_dataframe.iterrows():
    featuresets2.append((gender_country_features_2(row[1].nombre_final,ro
w[1].pais)
                         ,row[1].genero))
```

In [92]:

```
#Vamos a probar con nuestros nombres:
arr_nombres = ['Elena','Miguel Angel', 'Jose Antonio'
               ,'Pablo', 'Pedro', 'Gemma', 'Jorge','Luis'
               ,'Javier','Emmanuele','Jose Miguel','Adrián','Ana']
arr_pais = np.repeat('ES',len(arr_nombres))

adivina = pd.DataFrame()
adivina['nombre'] = arr_nombres
adivina['pais'] = arr_pais

#for nombres in df_final.nombre_final:
print "Algoritmo 1:"
print "--------------------------"

for nombres in arr_nombres:
    print nombres , " : " ,classifier.classify(gender_features(nombres))

print
print "Algoritmo 2:"
print "--------------------------"
for row in adivina.iterrows():
    feature = (gender_country_features_2(row[1].nombre,row[1].pais))
    print  row[1].nombre ,  ":" , classifier_2.classify(feature)
```

```
Algoritmo 1:
--------------------------
Elena  :  female
Miguel Angel  :  male
Jose Antonio  :  male
Pablo  :  male
Pedro  :  male
Gemma  :  female
Jorge  :  female
Luis  :  male
Javier  :  male
Emmanuele  :  female
Jose Miguel  :  male
Adrián  :  male
Ana  :  female

Algoritmo 2:
--------------------------
Elena : female
Miguel Angel : male
Jose Antonio : male
Pablo : male
Pedro : male
Gemma : female
Jorge : male
Luis : male
Javier : male
Emmanuele : female
Jose Miguel : male
Adrián : male
Ana : female
```

BIBLIOGRAFÍA

http://www.nltk.org/book/ch06.html (http://www.nltk.org/book/ch06.html)

In [ ]: