In [28]:

```
%matplotlib inline
```

Data Exploration

1.- Librerias

In [29]:

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

#Para convertir los datos que son categóricos
import sklearn.preprocessing as pp

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

#Base maps -> mirar como instalarlo en la bibliografía al final del documen
to
#http://gnperdue.github.io/yak-shaving/osx/python/matplotlib/2014/05/01/bas
emap-toolkit.html
from mpl_toolkits.basemap import Basemap


#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

```
-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 [30]:

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

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

print df.columns
```

```
Index([u'ciudad', u'email_server', u'edad', u'genero', u'hora_visita', u'id
ioma', u'is_weekend', u'nombre_final', u'os', u'pais', u'rango_horario',
u'timezone', u'ua_browser_family', u'ua_device', u'ua_device_family', u'u
a_is_bot', u'ua_is_movile', u'ua_is_pc', u'ua_is_tablet', u'ua_is_tounch_ca
pable', u'ua_os_family', u'weekday'], dtype='object')
```

In [31]:

```
#vamos a poner el tipo de los datos:

df.ciudad = df.ciudad.astype('category')
df.email_server = df.email_server.astype('category')
df.edad = df.edad.astype('category')
df.genero = df.genero.astype('category')
df.hora_visita = df.hora_visita.astype('category')
df.idioma = df.idioma.astype('category')
df.is_weekend = df.is_weekend.astype('category')
df.nombre_final = df.nombre_final.astype('category')
df.os = df.os.astype('category')
df.pais = df.pais.astype('category')
df.rango_horario = df.rango_horario.astype('category')
df.timezone = df.timezone.astype('category')
df.ua_browser_family = df.ua_browser_family.astype('category')
df.ua_device = df.ua_device.astype('category')
df.ua_device_family = df.ua_device_family.astype('category')
df.ua_is_bot = df.ua_is_bot.astype('category')
df.ua_is_movile = df.ua_is_movile.astype('category')
df.ua_is_pc = df.ua_is_pc.astype('category')
df.ua_is_tablet = df.ua_is_tablet.astype('category')
df.ua_is_tounch_capable = df.ua_is_tounch_capable.astype('category')
df.ua_os_family = df.ua_os_family.astype('category')
df.weekday = df.weekday.astype('category')
```

4.- Univariate Analysis

- **name:** email_server
- **description:** Servidor de email
- **type:** discreta

In [32]:

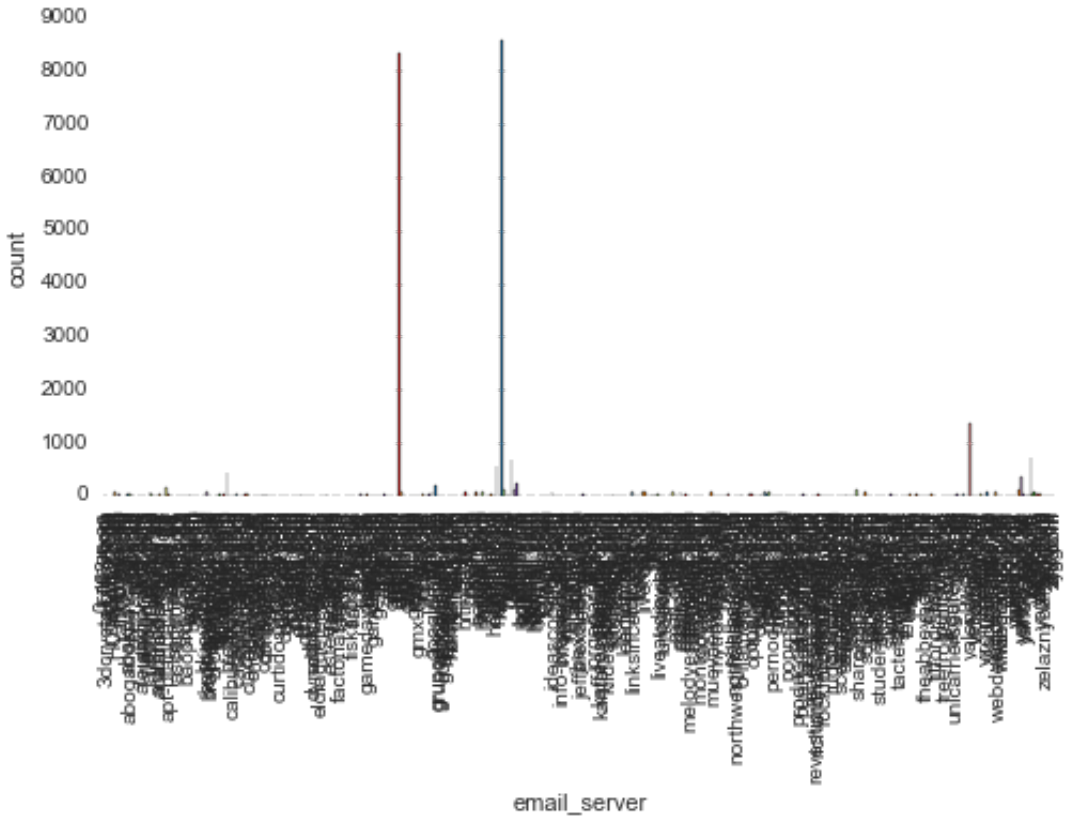
```
df.email_server.describe()
```

Out[32]:

```
count            26078
unique             520
top        hotmail.com
freq              8576
Name: email_server, dtype: object
```

In [33]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (30, 7)})
sns.barplot(df.email_server, palette="Paired");
```



Se ve claramente que hay 2 servidores que son los que más se usan y el resto se usan poco

In [34]:

```
#Porcentaje por servidor:

100.0*df.email_server.value_counts()/len(df.email_server)
```

Out[34]:

```
hotmail.com       32.885958
gmail.com         31.931130
vacio              5.307155
yahoo.es           2.822302
hotmail.es         2.584554
hotmail.co.uk      2.132065
btinternet.com     1.629726
yahoo.com          1.357466
hotmail.it         0.824450
googlemail.com     0.747757
aol.com            0.586701
seznam.cz          0.433315
hotmail.con        0.421811
yahoo.co.uk        0.375796
hotmail.fr         0.368126
...
karltatler.com          0.003835
kaffeeschluerfer.com    0.003835
kabelfoon.nl            0.003835
jubelsound.nl           0.003835
jsxxi.es                0.003835
jorgegarrido.com        0.003835
jmu.edu                 0.003835
jerez.es                0.003835
jbexclusivas.com        0.003835
islacanela.es           0.003835
ipsum.com               0.003835
infonegocio.com         0.003835
info-bremerhaven.de     0.003835
ideasconalma.com        0.003835
its.jnj.com             0.003835
Length: 520, dtype: float64
```

- **name:** hora_visita
- **description:** Hora en la que se conecta
- **type:** discreta

In [35]:

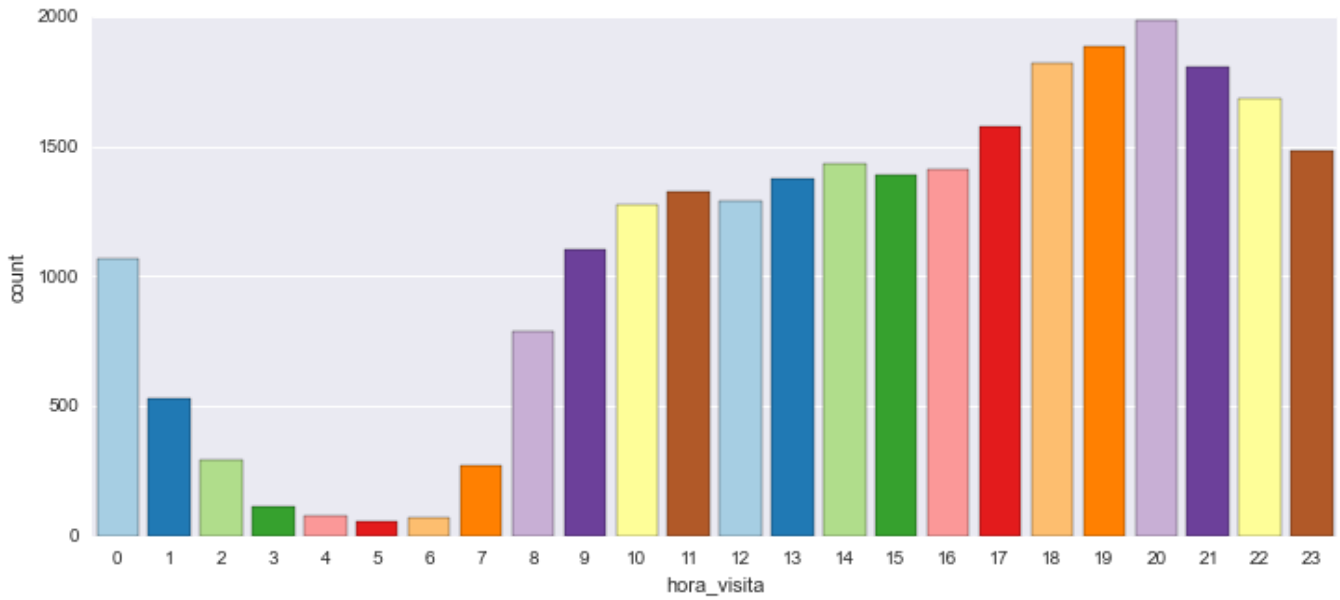
```
df.hora_visita.describe()
```

Out[35]:

```
count     26078
unique       24
top          20
freq       1987
Name: hora_visita, dtype: int64
```

In [36]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.hora_visita, palette="Paired");
```



In [37]:

```
#Porcentaje por hora:

100.0*df.hora_visita.value_counts()/len(df.hora_visita)
```

Out[37]:

```
20    7.619449
19    7.232150
18    6.971394
21    6.913874
22    6.449881
17    6.043408
23    5.690620
14    5.479715
16    5.418360
15    5.326329
13    5.257305
11    5.069407
12    4.927525
10    4.889179
9     4.229619
0     4.095406
8     3.025539
1     2.024695
2     1.119718
7     1.031521
3     0.421811
4     0.302937
6     0.260756
5     0.199402
dtype: float64
```

- **name:** is_weekend
- **description:** Es fin de semana cuando se conecta?
- **type:** discreta

In [38]:

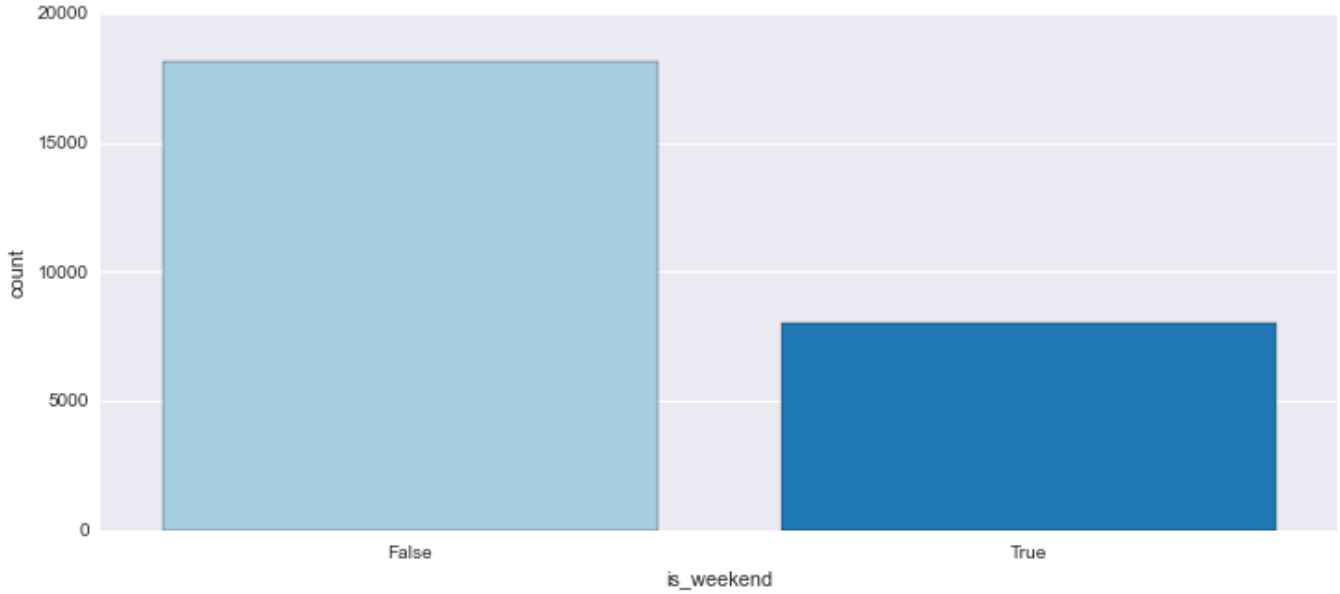
```
df.is_weekend.describe()
```

Out[38]:

```
count     26078
unique        2
top       False
freq      18090
Name: is_weekend, dtype: object
```

In [39]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.is_weekend, palette="Paired");
```



In [40]:

```
#Porcentaje por is_weekend:

100.0*df.is_weekend.value_counts()/len(df.is_weekend)
```

Out[40]:

```
False    69.368817
True     30.631183
dtype: float64
```

- **name:** os
- **description:** Sistema operativo del dispositivo
- **type:** discreta

In [41]:

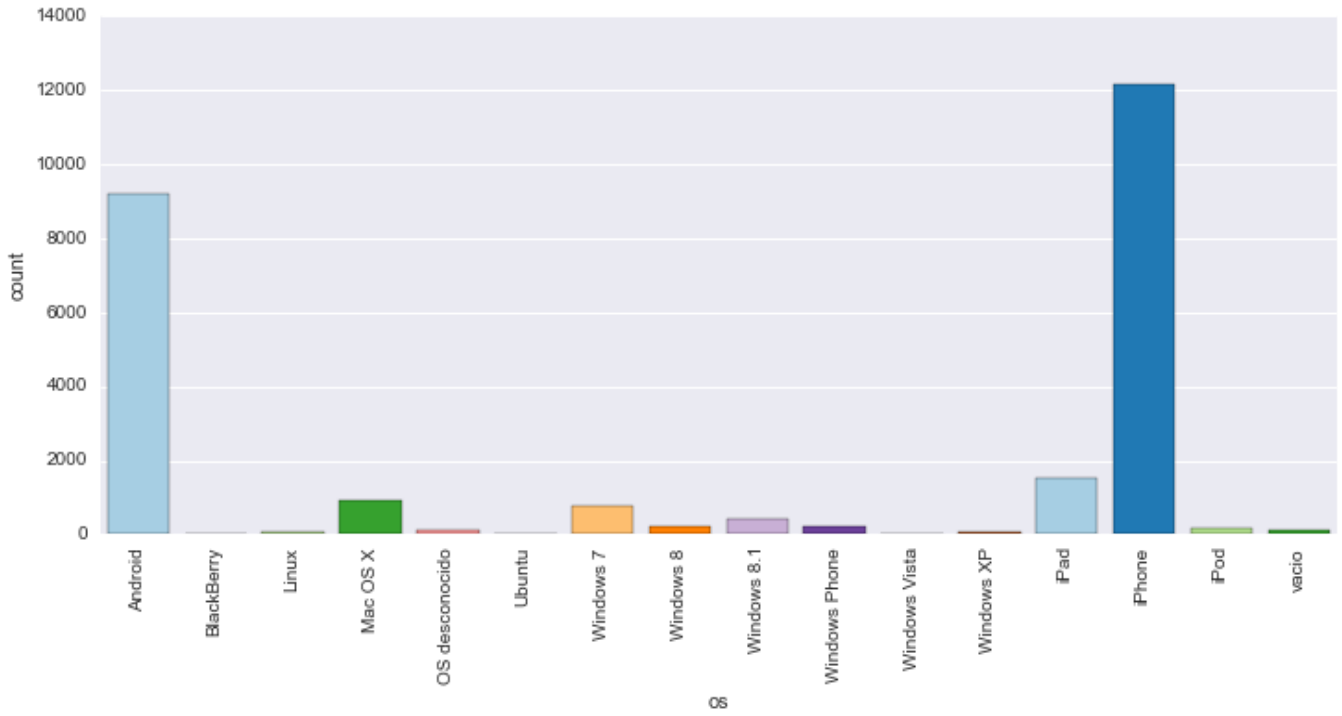
```
df.os.describe()
```

Out[41]:

```
count      26078
unique        16
top       iPhone
freq       12174
Name: os, dtype: object
```

In [42]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (12, 5)})
sns.barplot(df.os, palette="Paired");
```



In [43]:

```
#Porcentaje os:

100.0*df.os.value_counts()/len(df.os)
```

Out[43]:

```
iPhone            46.683028
Android           35.259606
iPad               5.905361
Mac OS X           3.524043
Windows 7          3.025539
Windows 8.1        1.694915
Windows 8          0.774599
Windows Phone      0.755426
iPod               0.694072
vacio              0.521512
OS desconocido     0.475497
Windows XP         0.260756
Linux              0.256922
Windows Vista      0.072858
BlackBerry         0.057520
Ubuntu             0.038346
dtype: float64
```

- **name:** pais
- **description:** pais
- **type:** discreta

In [44]:

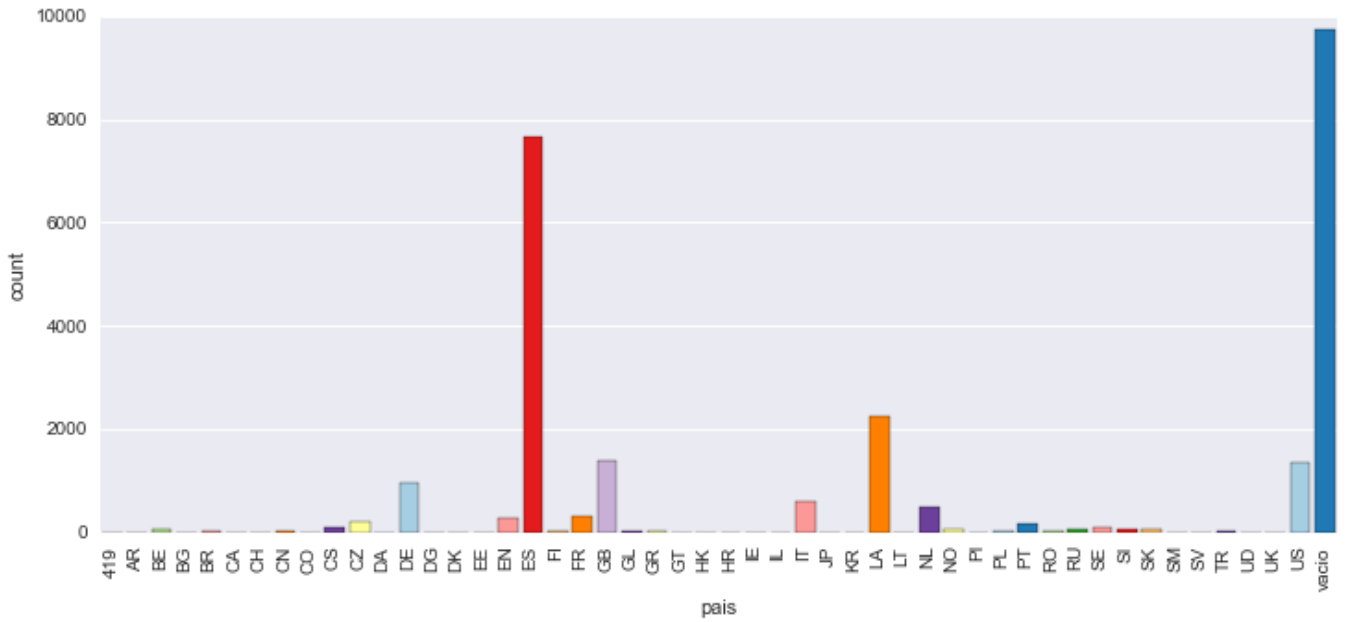
```
df.pais.describe()
```

Out[44]:

```
count     26078
unique       50
top       vacio
freq       9751
Name: pais, dtype: object
```

In [45]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (12, 5)})
sns.barplot(df.pais, palette="Paired");
```



In [46]:

```
#Porcentaje os:

100.0*df.pais.value_counts()/len(df.pais)
```

Out[46]:

```
vacio    37.391671
ES       29.331237
LA        8.616458
GB        5.322494
US        5.238132
DE        3.669760
IT        2.258609
NL        1.829128
FR        1.146560
EN        1.027686
CZ        0.736253
PT        0.659560
SE        0.444819
CS        0.333615
RU        0.268425
BE        0.218575
SI        0.214740
SK        0.199402
NO        0.180229
FI        0.107370
PL        0.099701
BR        0.095866
GR        0.095866
RO        0.069024
GL        0.069024
CN        0.061354
TR        0.049850
BG        0.030677
AR        0.026843
CO        0.023008
CA        0.019173
SV        0.019173
JP        0.015339
PI        0.015339
LT        0.015339
419       0.011504
EE        0.011504
IL        0.011504
DG        0.007669
DK        0.007669
HR        0.007669
IE        0.007669
CH        0.007669
DA        0.003835
GT        0.003835
KR        0.003835
SM        0.003835
UD        0.003835
UK        0.003835
HK        0.003835
dtype: float64
```

- **name:** rango horario
- **description:** Si se conecta por la mañana, mediodia ...
- **type:** discreta

In [47]:

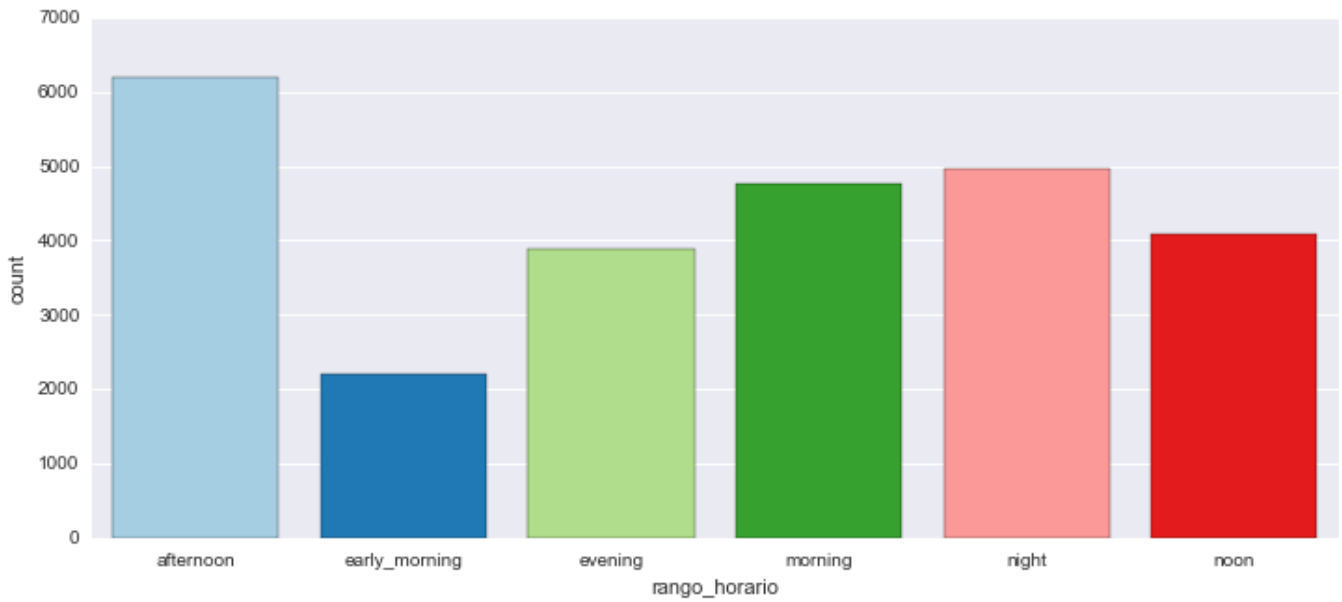
```
df.rango_horario.describe()
```

Out[47]:

```
count          26078
unique             6
top        afternoon
freq            6196
Name: rango_horario, dtype: object
```

In [48]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.rango_horario, palette="Paired");
```



In [49]:

```
#Porcentaje rango_horario:

100.0*df.rango_horario.value_counts()/len(df.pais)
```

Out[49]:

```
afternoon        23.759491
night            19.054375
morning          18.245264
noon             15.664545
evening          14.851599
early_morning     8.424726
dtype: float64
```

- **name:** time_zone
- **description:** zona_horaria (GMT + ??)
- **type:** discreta

In [50]:

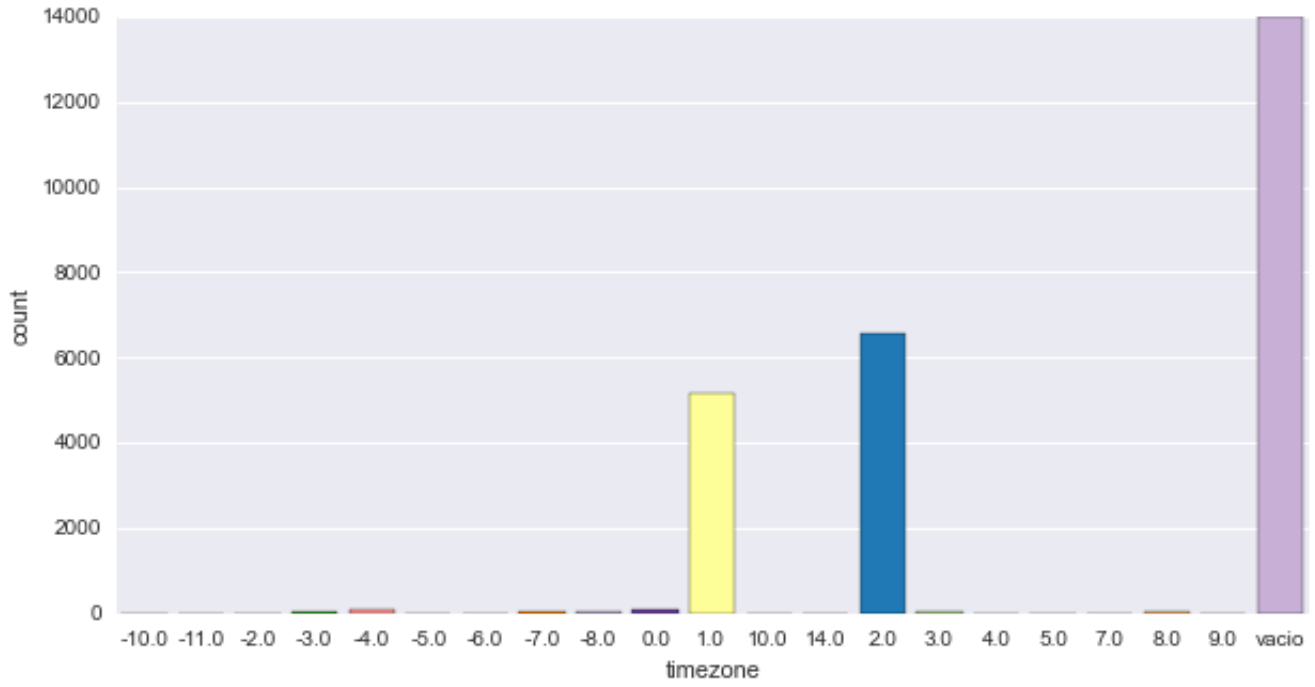
```
df.timezone.describe()
```

Out[50]:

```
count     26078
unique       21
top       vacio
freq      13982
Name: timezone, dtype: object
```

In [51]:

```
sns.set(rc={"figure.figsize": (10,5)})
sns.barplot(df.timezone, palette="Paired");
```



In [52]:

```
#Porcentaje rango_horario:

100.0*df.timezone.value_counts()/len(df.timezone)
```

Out[52]:

```
vacio    53.616075
2.0      25.247335
1.0      19.752282
-4.0      0.352788
0.0       0.318276
3.0       0.184063
-3.0      0.153386
-7.0      0.138047
-8.0      0.065189
8.0       0.061354
-5.0      0.038346
4.0       0.019173
-6.0      0.011504
5.0       0.011504
7.0       0.007669
9.0       0.003835
10.0      0.003835
14.0      0.003835
-2.0      0.003835
-11.0     0.003835
-10.0     0.003835
dtype: float64
```

- **name:** ua_browser_family
- **description:** browser_family que aparece en el user_agent
- **type:** discreta

In [53]:

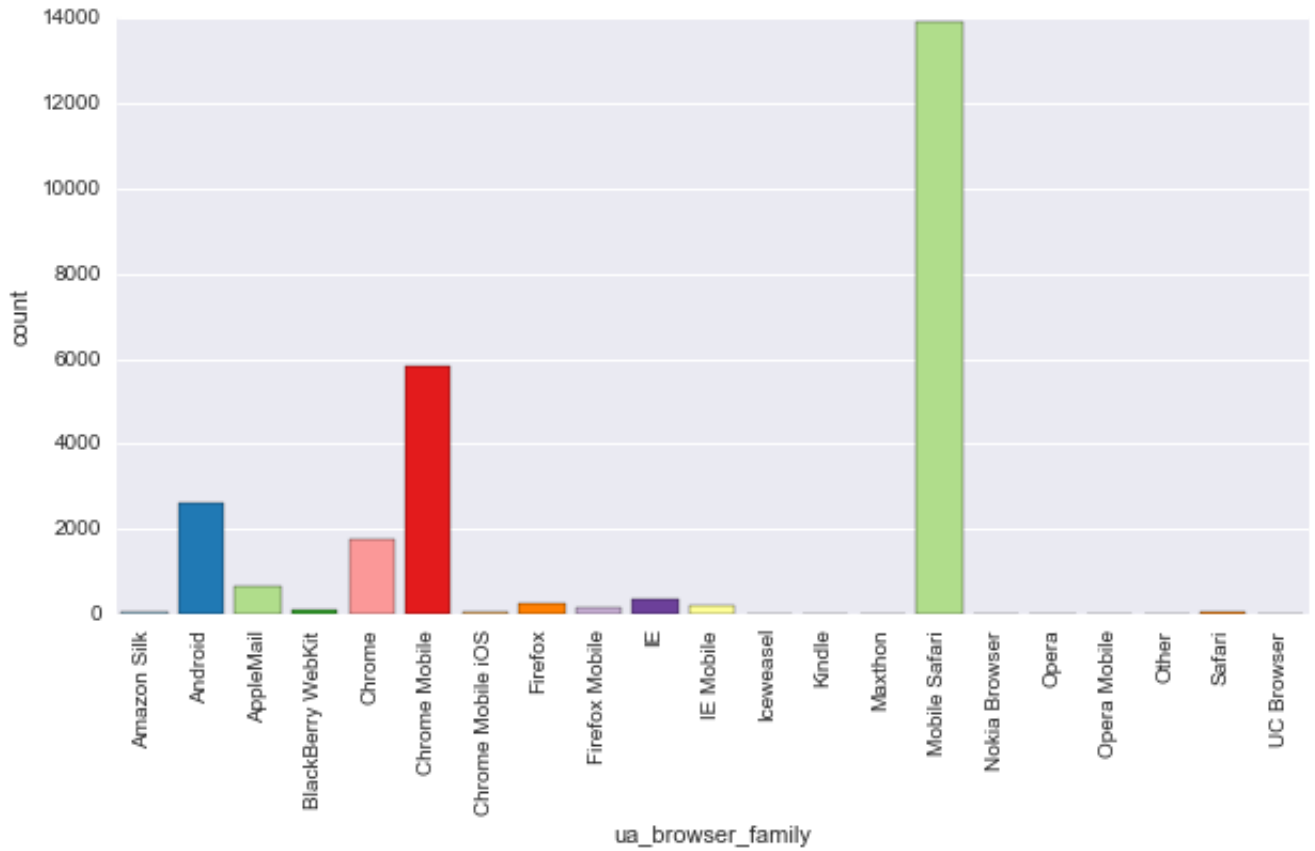
```
df.ua_browser_family.describe()
```

Out[53]:

```
count              26078
unique                21
top       Mobile Safari
freq               13886
Name: ua_browser_family, dtype: object
```

In [54]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (10,5)})
sns.barplot(df.ua_browser_family, palette="Paired");
```



In [55]:

```
#Porcentaje rango_horario:

100.0*df.ua_browser_family.value_counts()/len(df.ua_browser_family)
```

Out[55]:

```
Mobile Safari        53.247948
Chrome Mobile        22.363678
Android              10.096633
Chrome                6.760488
AppleMail             2.515530
IE                    1.365135
Firefox               1.016182
IE Mobile             0.766930
Firefox Mobile        0.533016
BlackBerry WebKit     0.463993
Safari                0.287599
Chrome Mobile iOS     0.199402
Amazon Silk           0.161055
Opera                 0.069024
Kindle                0.053685
Opera Mobile          0.038346
Maxthon               0.034512
Iceweasel             0.007669
Nokia Browser         0.007669
UC Browser            0.007669
Other                 0.003835
dtype: float64
```

- **name:** ua_device
- **description:** dispositivo en el user_agent
- **type:** discreta

In [56]:

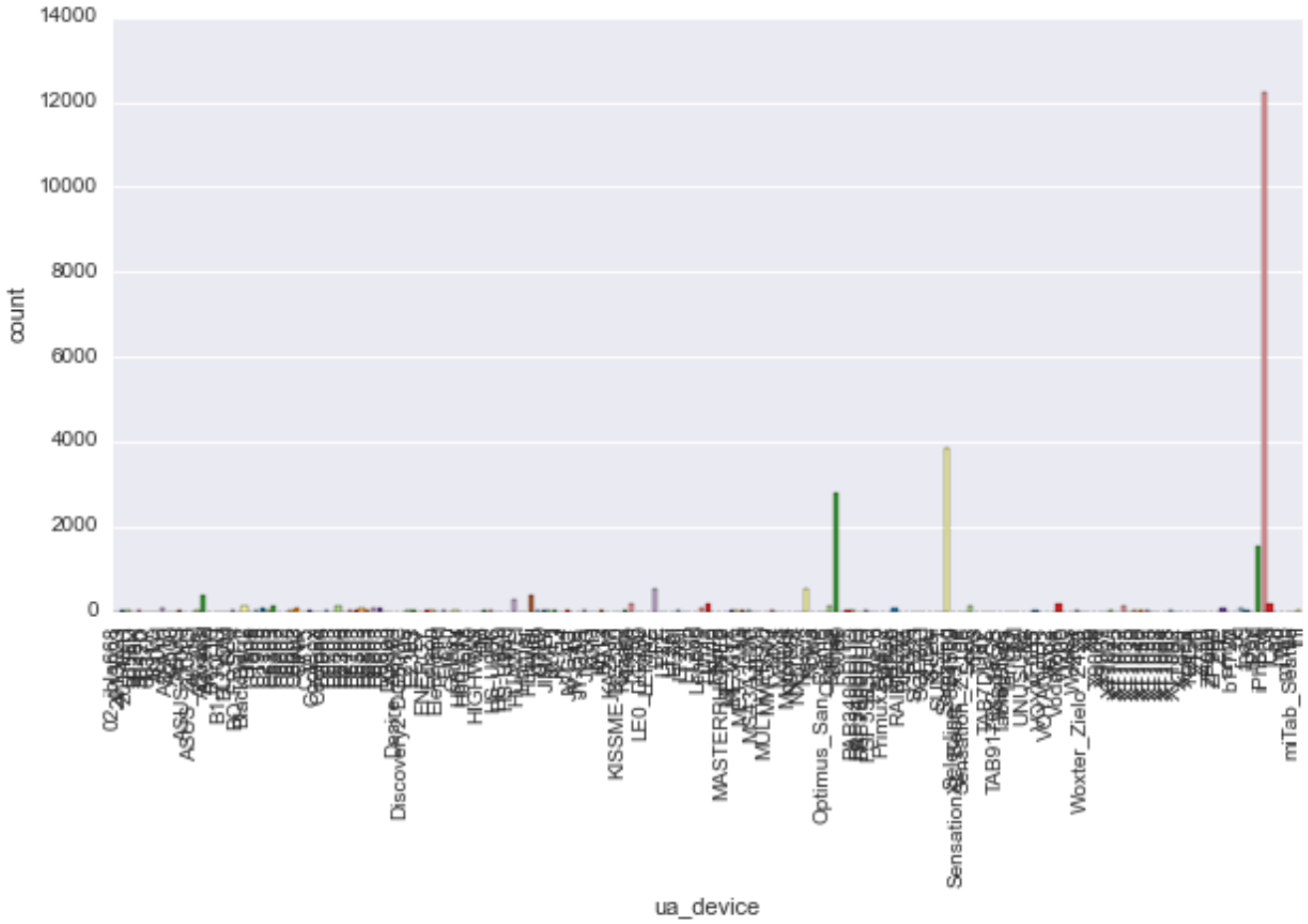
```
df.ua_device.describe()
```

Out[56]:

```
count      26078
unique       203
top       iPhone
freq       12214
Name: ua_device, dtype: object
```

In [57]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (18,5)})
sns.barplot(df.ua_device, palette="Paired");
```



In [58]:

```
#Porcentaje device :

100.0*df.ua_device.value_counts()/len(df.ua_device)
```

Out[58]:

```
iPhone      46.836414
Samsung     14.686709
Other       10.625815
iPad         5.916865
Nexus        2.101388
LG           1.967175
Aquaris      1.491679
Huawei       1.395813
HTC          1.150395
Kindle       0.743922
Lumia        0.713245
iPod         0.694072
Vodafone     0.625048
Sony         0.552190
Orange       0.475497
...
W200                 0.003835
HS-U980              0.003835
W8s                  0.003835
DG800                0.003835
Woxter_Zielo_Z420    0.003835
X6                   0.003835
X9                   0.003835
X9006                0.003835
X9007                0.003835
XT1025               0.003835
D2306                0.003835
LT22i                0.003835
XT1053               0.003835
SLIDE                0.003835
PAP5400DUO           0.003835
Length: 203, dtype: float64
```

- **name:** ua_device
- **description:** dispositivo en el user_agent
- **type:** discreta

In [59]:

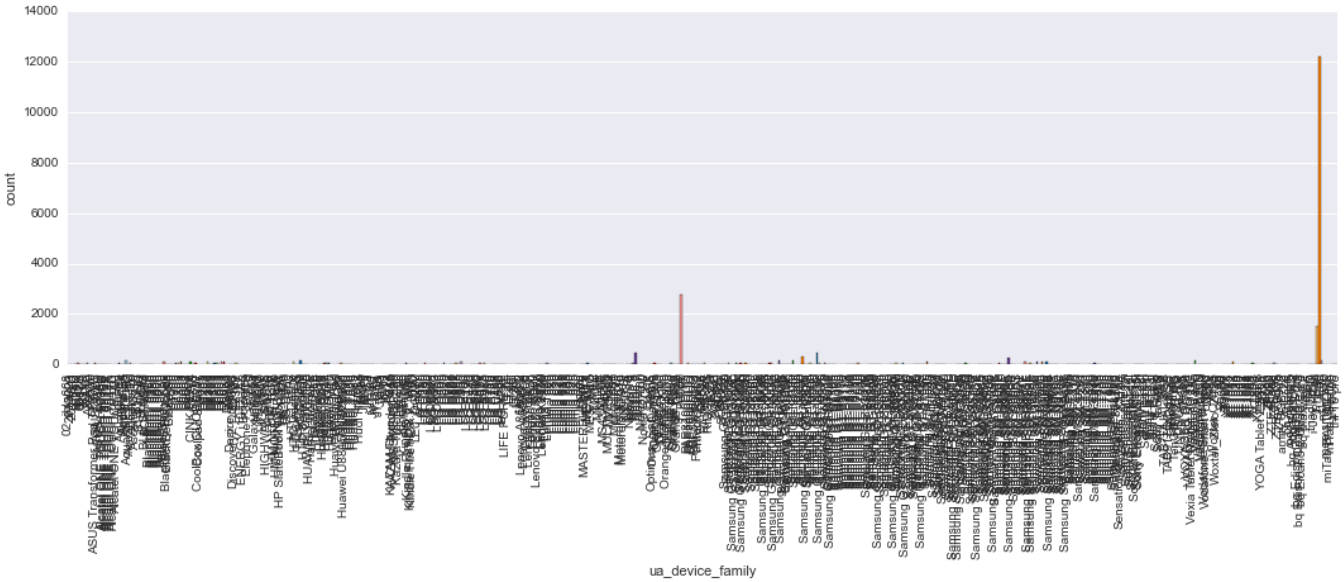
```
df.ua_device_family.describe()
```

Out[59]:

```
count      26078
unique       531
top       iPhone
freq       12214
Name: ua_device_family, dtype: object
```

In [60]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (18,5)})
sns.barplot(df.ua_device_family, palette="Paired");
```



In [61]:

```
#Porcentaje device :

100.0*df.ua_device_family.value_counts()/len(df.ua_device_family)
```

Out[61]:

```
iPhone              46.836414
Other               10.625815
iPad                 5.916865
Nexus 5              1.752435
Samsung GT-I9505     1.687246
Samsung GT-I9300     1.150395
Samsung SM-G900F     1.069867
iPod                 0.694072
Aquaris E5           0.678733
Samsung GT-I9195     0.655725
HTC One              0.628883
Vodafone 785         0.575197
Samsung GT-I9100     0.544520
Samsung SM-N9005     0.513843
C5303                0.471662
...
Samsung GT-I8730            0.003835
Samsung GT-I9070P-ORANGE    0.003835
Samsung GT-I9105P           0.003835
Hudl HT7S3                  0.003835
Samsung GT-I9205            0.003835
Samsung GT-N7000-ORANGE     0.003835
Samsung GT-S5360            0.003835
Huawei U8815NC02B891        0.003835
Samsung GT-N7105            0.003835
Samsung GT-P3100            0.003835
Samsung GT-P3113            0.003835
Huawei MT7                  0.003835
Huawei G700                 0.003835
Samsung GT-S5300            0.003835
Samsung SGH-T999            0.003835
Length: 531, dtype: float64
```

- **name:** ua_is_bot
- **description:** si el dispositivo es un robot
- **type:** discreta

In [62]:

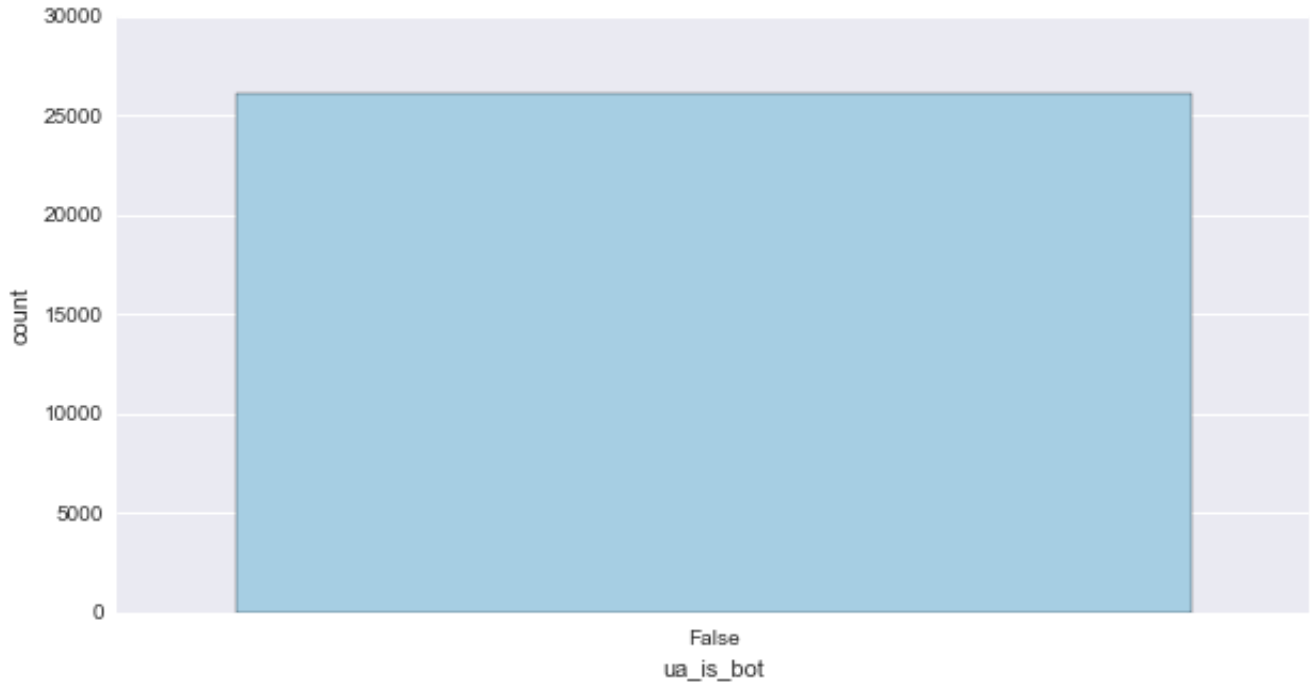
```
df.ua_is_bot.describe()
```

Out[62]:

```
count     26078
unique        1
top       False
freq      26078
Name: ua_is_bot, dtype: object
```

In [63]:

```
sns.set(rc={"figure.figsize": (10,5)})
sns.barplot(df.ua_is_bot, palette="Paired");
```



Como vemos esta variable no aporta nada al modelo, de hecho, no tiene sentido que haya conexiones de tipo robot

---

- **name:** ua_is_movile
- **description:** si el dispositivo es un movil
- **type:** discreta

In [64]:

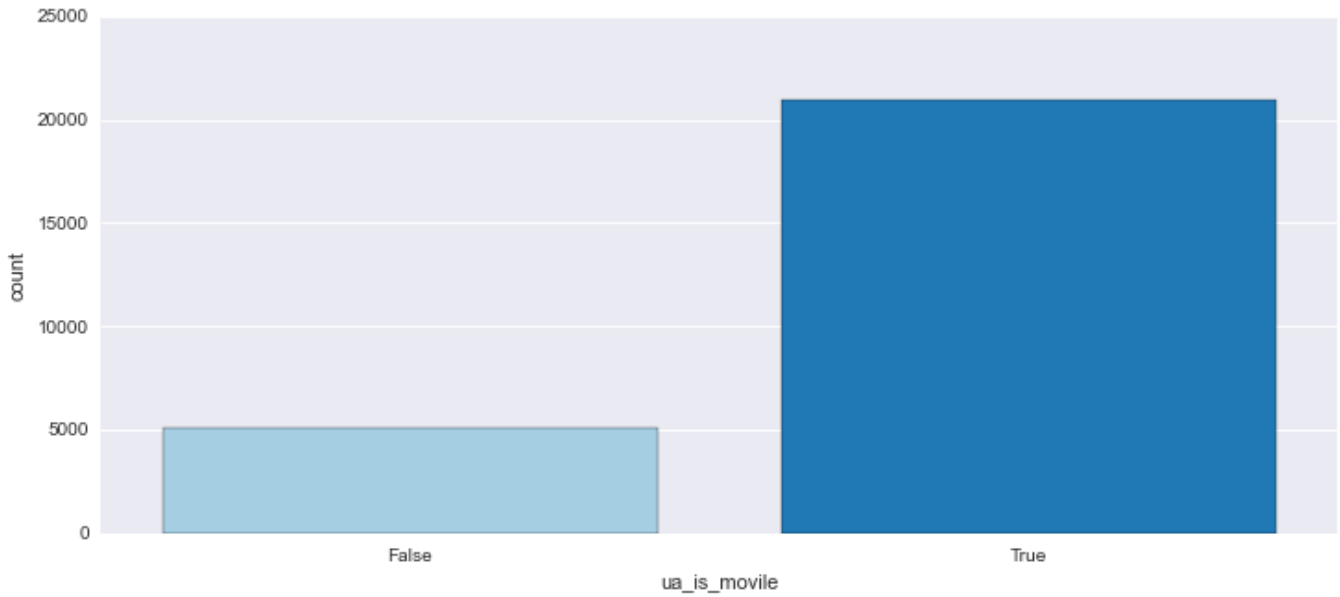
```
df.ua_is_movile.describe()
```

Out[64]:

```
count     26078
unique        2
top        True
freq      20948
Name: ua_is_movile, dtype: object
```

In [65]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.ua_is_movile, palette="Paired");
```



In [66]:

```
#Porcentaje:

100.0*df.ua_is_movile.value_counts()/len(df.ua_is_movile)
```

Out[66]:

```
True     80.328246
False    19.671754
dtype: float64
```

- **name:** ua_is_pc
- **description:** si el dispositivo es un pc
- **type:** discreta

In [67]:

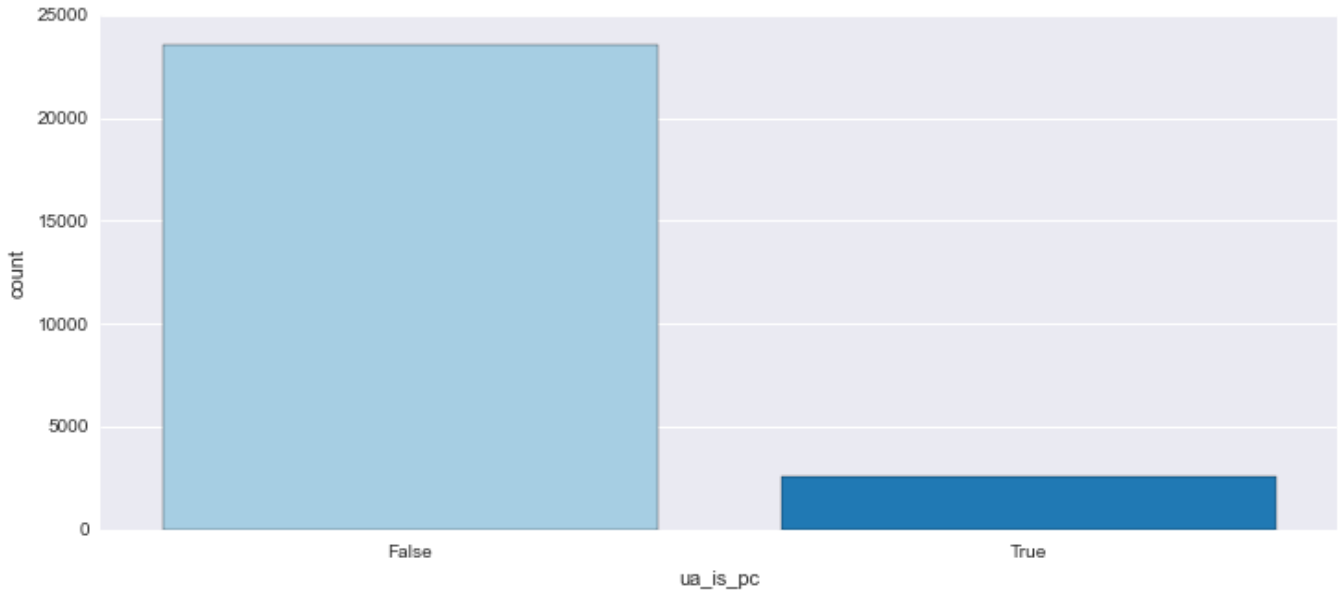
```
df.ua_is_pc.describe()
```

Out[67]:

```
count     26078
unique        2
top       False
freq      23504
Name: ua_is_pc, dtype: object
```

In [68]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.ua_is_pc, palette="Paired");
```



In [69]:

```
#Porcentaje:

100.0*df.ua_is_pc.value_counts()/len(df.ua_is_pc)
```

Out[69]:

```
False    90.129611
True      9.870389
dtype: float64
```

- **name:** ua_is_tablet
- **description:** si el dispositivo es una tablet
- **type:** discreta

In [70]:

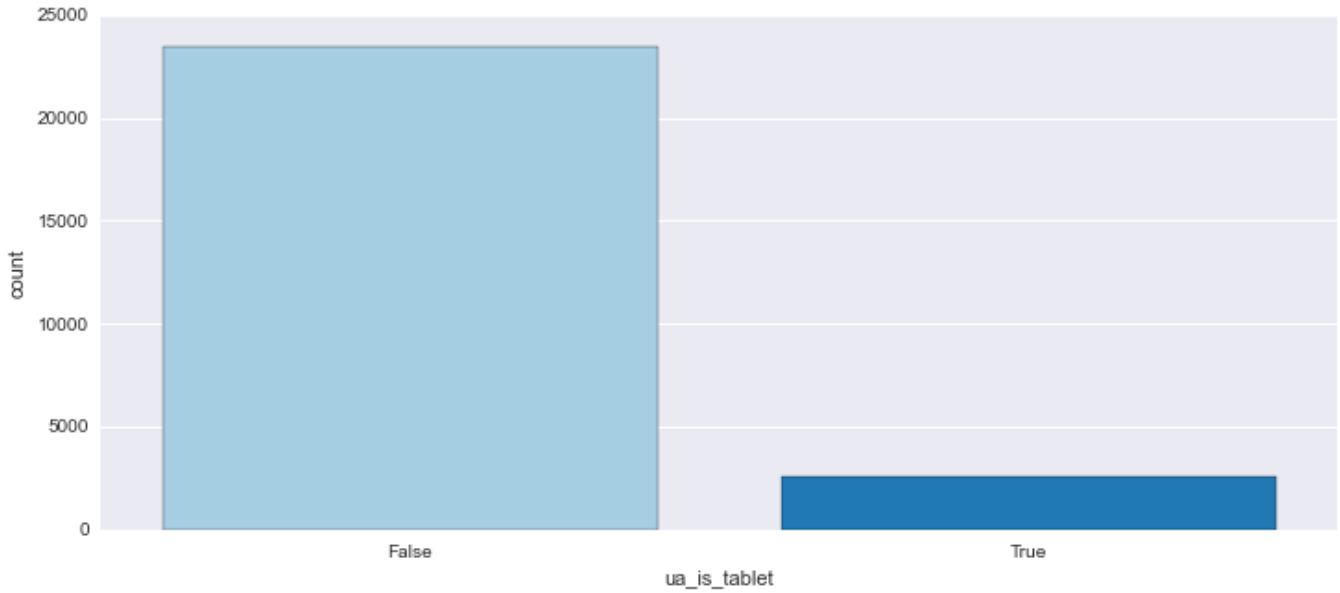
```
df.ua_is_tablet.describe()
```

Out[70]:

```
count     26078
unique        2
top       False
freq      23492
Name: ua_is_tablet, dtype: object
```

In [71]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.ua_is_tablet, palette="Paired");
```



In [72]:

```
#Porcentaje :

100.0*df.ua_is_tablet.value_counts()/len(df.ua_is_tablet)
```

Out[72]:

```
False    90.083595
True      9.916405
dtype: float64
```

- **name:** ua_is_touch_capable
- **description:** si el dispositivo es un táctil
- **type:** discreta

In [73]:

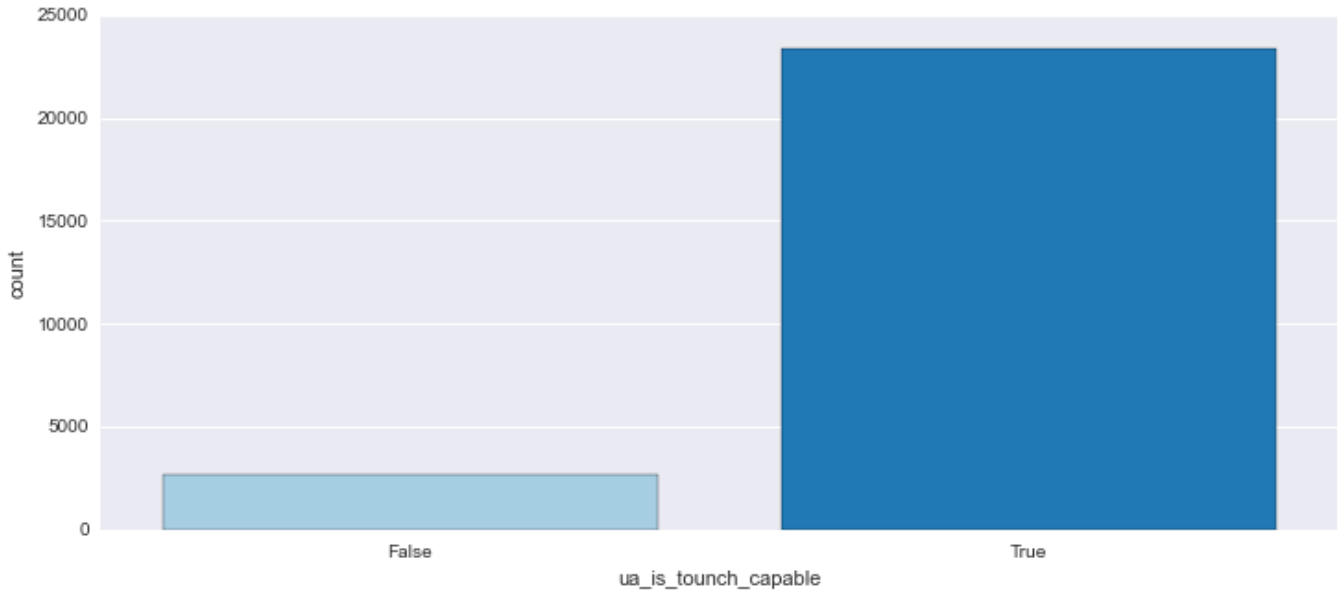
```
df.ua_is_tounch_capable.describe()
```

Out[73]:

```
count     26078
unique        2
top        True
freq      23399
Name: ua_is_tounch_capable, dtype: object
```

In [74]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.ua_is_tounch_capable, palette="Paired");
```



In [75]:

```
#Porcentaje device :

100.0*df.ua_is_tounch_capable.value_counts()/len(df.ua_is_tounch_capable)
```

Out[75]:

```
True     89.726973
False    10.273027
dtype: float64
```

- **name:** ua_os_family
- **description:** os family del user agent
- **type:** discreta

In [76]:

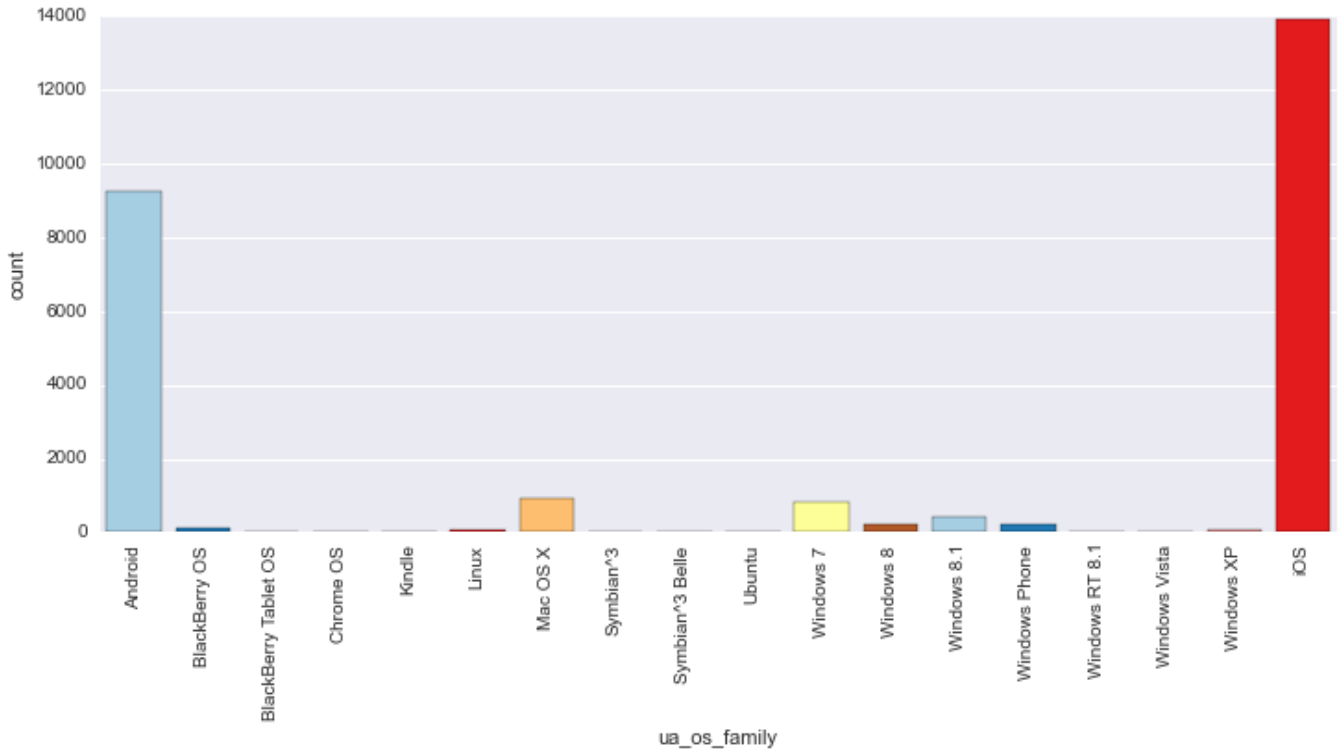
```
df.ua_os_family.describe()
```

Out[76]:

```
count     26078
unique       18
top         iOS
freq      13938
Name: ua_os_family, dtype: object
```

In [77]:

```
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.ua_os_family, palette="Paired");
```



In [78]:

```
#Porcentaje:

100.0*df.ua_os_family.value_counts()/len(df.ua_os_family)
```

Out[78]:

```
iOS                    53.447350
Android                35.455173
Mac OS X                3.535547
Windows 7               3.106066
Windows 8.1             1.652734
Windows 8               0.774599
Windows Phone           0.766930
BlackBerry OS           0.460158
Windows XP              0.260756
Linux                   0.256922
Windows Vista           0.072858
Chrome OS               0.061354
Kindle                  0.053685
Windows RT 8.1          0.046016
Ubuntu                  0.038346
Symbian^3               0.003835
BlackBerry Tablet OS    0.003835
Symbian^3 Belle         0.003835
dtype: float64
```

- **name:** weekday
- **description:** dia de la semana
- **type:** discreta

In [79]:

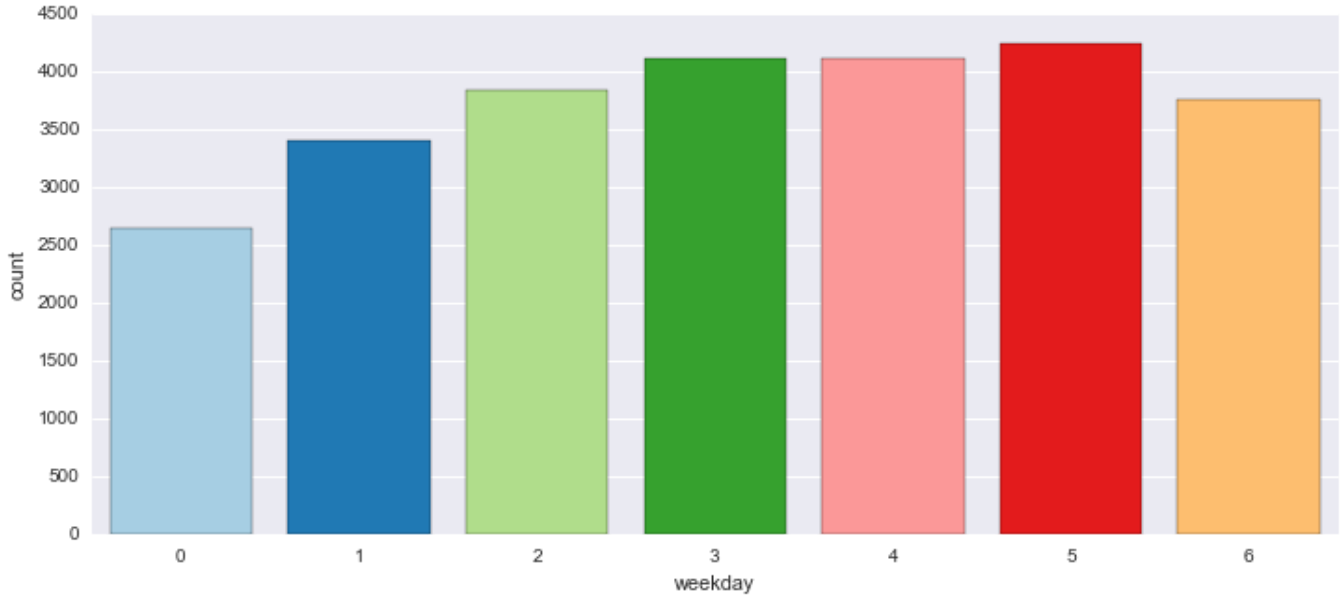
```
df.weekday.describe()
```

Out[79]:

```
count     26078
unique        7
top           5
freq       4236
Name: weekday, dtype: int64
```

In [80]:

```
sns.set(rc={"figure.figsize": (12,5)})
sns.barplot(df.weekday, palette="Paired");
```



In [81]:

```
#Porcentaje:

100.0*df.weekday.value_counts()/len(df.weekday)
```

Out[81]:

```
5    16.243577
3    15.764246
4    15.745072
2    14.679040
6    14.387606
1    13.037810
0    10.142649
dtype: float64
```

6.- Guardamos a csv para poder seguir con el proceso

In [82]:

```
df.to_csv('../csv/datos_explorados.csv')
```