Pandas bowling: convierte tus datos en información

Introducción a la manipulación de datos utilizando pandas contra un set de datos públicos. Data munging: filtering, merging, grouping, estadísticas comunes e introducción al plotting.

Pandas bowling

In [1]:
```
import pandas as pd
import csv
import re
import unicodedata
```

In [2]:
```
# Delimiter ;
# Awkward encoding

dialect = csv.excel()
dialect.delimiter = ';'
# Source: http://www.datosabiertos.jcyl.es/web/jcyl/risp/es/directorio/ba
res/1284211832884.csv
bars = pd.read_csv('bares.csv', dialect=dialect, encoding='cp1252')

to_ascii = lambda text: unicodedata.normalize('NFKD', text).encode('ASCII
', 'ignore').upper()
```

In [3]:
```
bars
```

Out[3]:
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13984 entries, 0 to 13983
```

```
Data columns (total 15 columns):
Nombre          13984  non-null values
Dirección       13915  non-null values
C.Postal        13958  non-null values
Provincia       13984  non-null values
Municipio       13984  non-null values
Localidad       13984  non-null values
Nucleo          9786  non-null values
Teléfono 1      11849  non-null values
Teléfono 2      658  non-null values
Teléfono 3      21  non-null values
Fax             45  non-null values
Email           255  non-null values
web             39  non-null values
Q Calidad       0  non-null values
Unnamed: 14     0  non-null values
dtypes: float64(2), object(13)
```

In [4]:
```
bars.tail(1)
```

Out[4]:

	Nombre	Dirección	C.Postal	Provincia	Municipio	Localidad	Nucle
13983	ARCO DE TRIUNFO	VILLAR Y MACIAS, 7	37003	Salamanca	Salamanca	SALAMANCA	SALAM

In [5]:
```
bars.Localidad.apply(lambda x: x.find('LA ') != -1)
```

Out[5]:
```
0      False
1      False
2      False
3      False
4      False
5      False
6      False
7      False
8      False
9      False
10     False
11     False
12     False
13     False
14     False
...
```

```
13969    False
13970    False
13971    False
13972    False
13973     True
13974    False
13975    False
13976    False
13977    False
13978    False
13979    False
13980    False
13981    False
13982    False
13983    False
Name: Localidad, Length: 13984, dtype: bool
```

In [6]:
```
bars[bars.Localidad.apply(lambda x: x.find('LA ') != -1)]
```

Out[6]:
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 805 entries, 46 to 13973
Data columns (total 15 columns):
Nombre          805  non-null values
Dirección       795  non-null values
C.Postal        803  non-null values
Provincia       805  non-null values
Municipio       805  non-null values
Localidad       805  non-null values
Nucleo          574  non-null values
Teléfono 1      685  non-null values
Teléfono 2      51  non-null values
Teléfono 3      0  non-null values
Fax             3  non-null values
Email           17  non-null values
web             1  non-null values
Q Calidad       0  non-null values
Unnamed: 14     0  non-null values
dtypes: float64(2), object(13)
```

In [7]:
```
# Selección de filas
bars.iloc[0]
bars.loc[0]
bars.ix[0]
bars.T[0]
```

Out[7]:
```
Nombre                CASA PEDRO
Dirección       NUÑEZ DE ARCE 4
```

```
C.Postal                 47002
Provincia           Valladolid
Municipio           Valladolid
Localidad           VALLADOLID
Nucleo              VALLADOLID
Teléfono 1           983000000
Teléfono 2                 NaN
Teléfono 3                 NaN
Fax                        NaN
Email                      NaN
web                        NaN
Q Calidad                  NaN
Unnamed: 14                NaN
Name: 0, dtype: object
```

In [8]:
```
bars[0]
```

```
---------------------------------------------------------------------------
--
KeyError                                  Traceback (most recent call las
t)
<ipython-input-8-4cadcae3962b> in <module>()
----> 1 bars[0]

//anaconda/lib/python2.7/site-packages/pandas/core/frame.pyc in __getitem
__(self, key)
   2001             # get column
   2002             if self.columns.is_unique:
-> 2003                 return self._get_item_cache(key)
   2004 
   2005             # duplicate columns

//anaconda/lib/python2.7/site-packages/pandas/core/generic.pyc in _get_it
em_cache(self, item)
    665             return cache[item]
    666         except Exception:
--> 667             values = self._data.get(item)
    668             res = self._box_item_values(item, values)
    669             cache[item] = res

//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in get(s
elf, item)
   1653     def get(self, item):
   1654         if self.items.is_unique:
-> 1655             _, block = self._find_block(item)
   1656             return block.get(item)
   1657         else:
```

```
//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in _find
_block(self, item)
   1933 
   1934     def _find_block(self, item):
-> 1935         self._check_have(item)
   1936         for i, block in enumerate(self.blocks):
   1937             if item in block:

//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in _chec
k_have(self, item)
   1940     def _check_have(self, item):
   1941         if item not in self.items:
-> 1942             raise KeyError('no item named %s' % com.pprint_thing(
item))
   1943 
   1944     def reindex_axis(self, new_axis, method=None, axis=0, copy=Tr
ue):

KeyError: u'no item named 0'
```

In [ ]:
```
bars.tail(1)
```

In [ ]:
```
web_only = bars.dropna(subset=['web'])
len(web_only)
web_only[web_only.Localidad == 'BURGOS']
```

In [ ]:
```
bars["C.Postal"]
```

In [ ]:
```
bars.Provincia
```

In [9]:
```
by_provincia_gb = bars.groupby('Provincia')
by_provincia_gb
```

Out[9]:
```
<pandas.core.groupby.DataFrameGroupBy object at 0x108cf5f50>
```

In [10]:
```
by_provincia_gb.web.get_group(u'León').dropna()
```

Out[10]:
```
11406    seaki.com
Name: web, dtype: object
```

In [11]:
```
by_provincia = pd.DataFrame({'Bares': by_provincia_gb.size()})
by_provincia
```

Out[11]:

	Bares

Provincia	
Burgos	1796
León	2224
Palencia	1057
Salamanca	1341
Segovia	804
Soria	353
Valladolid	3724
Zamora	1282
Ávila	1403

```
In [12]: by_provincia.index
```

```
Out[12]: Index([u'Burgos', u'León', u'Palencia', u'Salamanca', u'Segovia', u'Soria
         ', u'Valladolid', u'Zamora', u'Ávila'], dtype=object)
```

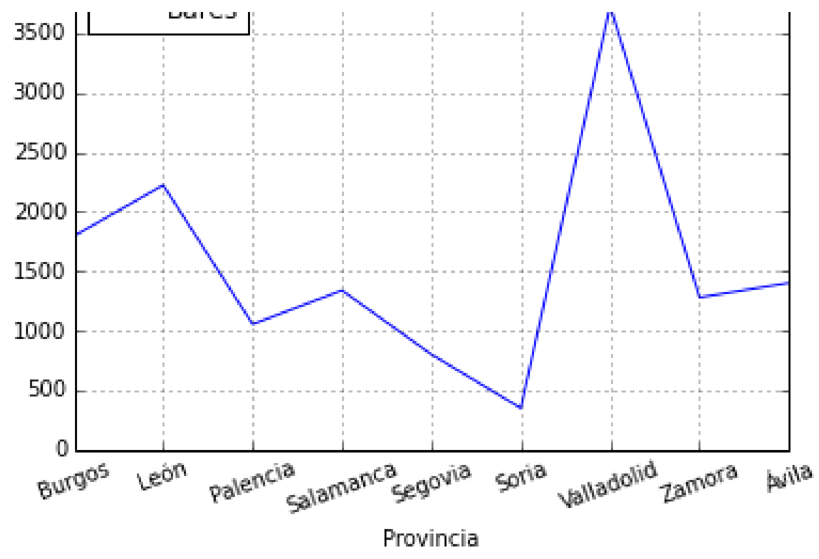
```
In [13]: by_provincia.sort('Bares', ascending=False)
```

Out[13]:

	Bares
Provincia	
Valladolid	3724
León	2224
Burgos	1796
Ávila	1403
Salamanca	1341
Zamora	1282
Palencia	1057
Segovia	804
Soria	353

```
In [14]: by_provincia.plot(rot=20)
```

```
Out[14]: <matplotlib.axes.AxesSubplot at 0x108d913d0>
```



In [15]:
```
population = pd.read_excel('poblacion_por_provincias.xls', 'poblacion')
population['Provincia'] = population.Provincia.apply(to_ascii)
population.columns = 'Provincia', 'Población'
population = population.set_index('Provincia')
population
```

Out[15]:

	Población
Provincia	
AVILA	169505
BURGOS	367906
LEON	488991
PALENCIA	168721
SALAMANCA	347377
SEGOVIA	161640
SORIA	93389
VALLADOLID	530590
ZAMORA	189037

In [16]:
```
joined = population.join(by_provincia)
joined
```

Out[16]:

	Población	**Bares**
Provincia		
AVILA	169505	NaN
BURGOS	367906	NaN

LEON	488991	NaN
PALENCIA	168721	NaN
SALAMANCA	347377	NaN
SEGOVIA	161640	NaN
SORIA	93389	NaN
VALLADOLID	530590	NaN
ZAMORA	189037	NaN

In [17]:
```
by_provincia = by_provincia.set_index(by_provincia.index.map(lambda prov:
 to_ascii(prov)))
```

In [18]:
```
joined = population.join(by_provincia)
joined
```

Out[18]:

	Población	**Bares**
Provincia		
AVILA	169505	1403
BURGOS	367906	1796
LEON	488991	2224
PALENCIA	168721	1057
SALAMANCA	347377	1341
SEGOVIA	161640	804
SORIA	93389	353
VALLADOLID	530590	3724
ZAMORA	189037	1282

In [19]:
```
joined['Población'].sum()
```

Out[19]:
```
2517156.0
```

In [20]:
```
joined.Bares.std()
```

Out[20]:
```
975.3478581739156
```

In [21]:
```
joined[['Población', 'Bares']].corr()
```

Out[21]:

	Población	Bares
Población	1.000000	0.890913
Bares	0.890913	1.000000

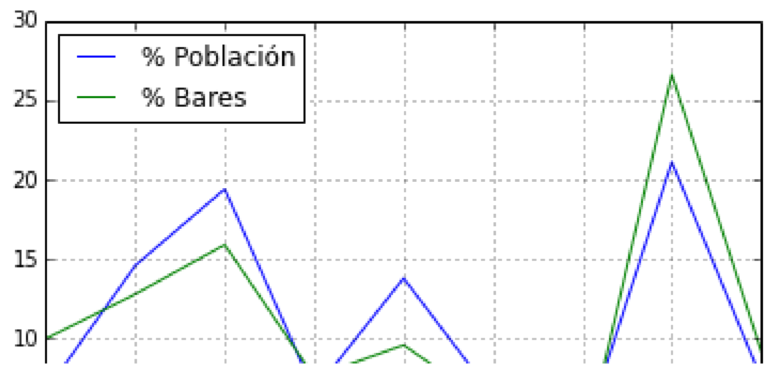
In [22]:
```
joined['Habitantes por bar'] = joined['Población'].astype(float) / joined
.Bares
joined['Habitantes por bar'] = joined['Habitantes por bar'].apply(round)
joined['% Bares'] = joined.Bares.apply(lambda number: round(100 * number
/ joined.Bares.sum().astype(float), 1))
joined['% Población'] = joined['Población'].apply(lambda people: round(10
0 * people / joined['Población'].sum().astype(float), 1))
joined
```

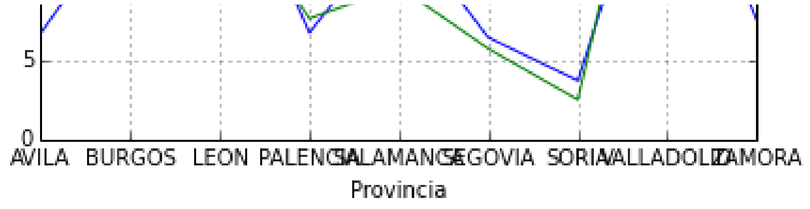
Out[22]:

	Población	Bares	Habitantes por bar	% Bares	% Población
Provincia					
AVILA	169505	1403	121	10.0	6.7
BURGOS	367906	1796	205	12.8	14.6
LEON	488991	2224	220	15.9	19.4
PALENCIA	168721	1057	160	7.6	6.7
SALAMANCA	347377	1341	259	9.6	13.8
SEGOVIA	161640	804	201	5.7	6.4
SORIA	93389	353	265	2.5	3.7
VALLADOLID	530590	3724	142	26.6	21.1
ZAMORA	189037	1282	147	9.2	7.5

In [23]:
```
joined[['% Población', '% Bares']].plot()
```

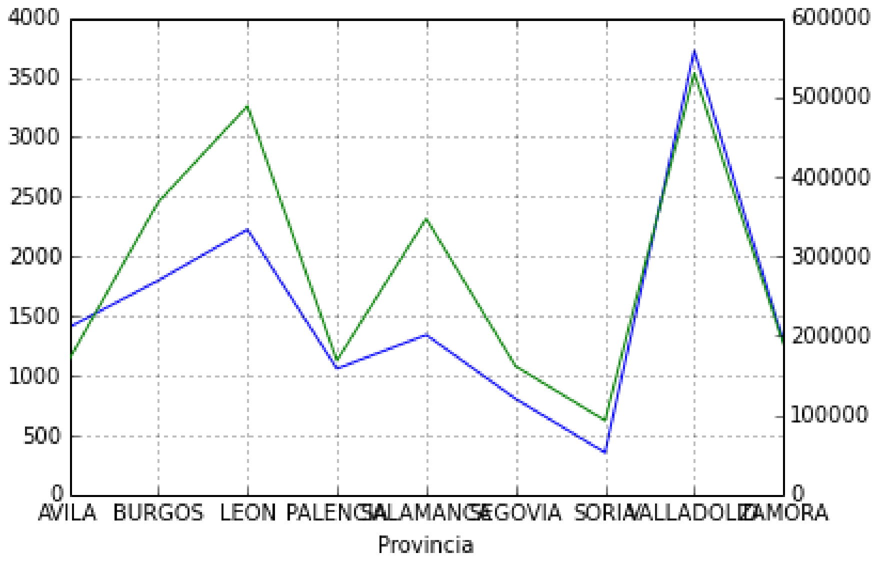
Out[23]:
```
<matplotlib.axes.AxesSubplot at 0x108d43650>
```





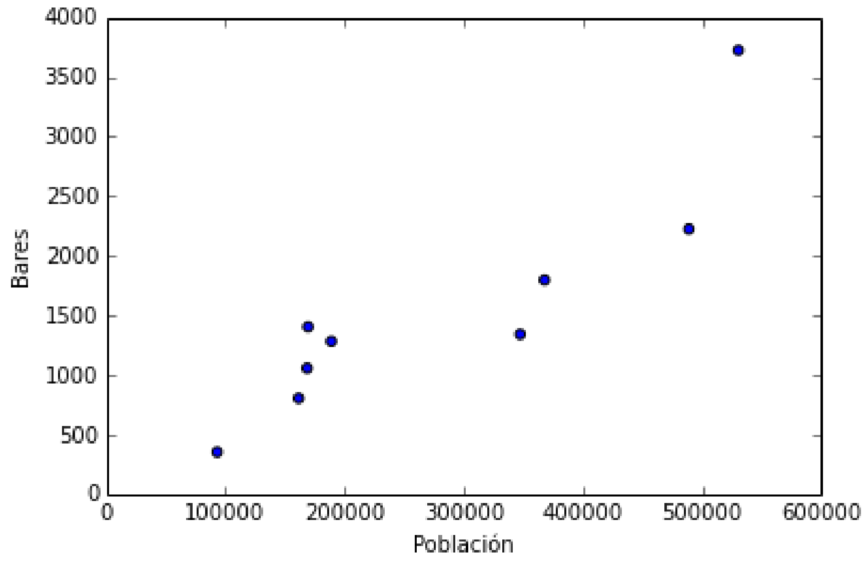
In [24]:
```
pob_bar = joined[['Población', 'Bares']]

pob_bar.Bares.plot()
pob_bar['Población'].plot(secondary_y=True, style='g')
```

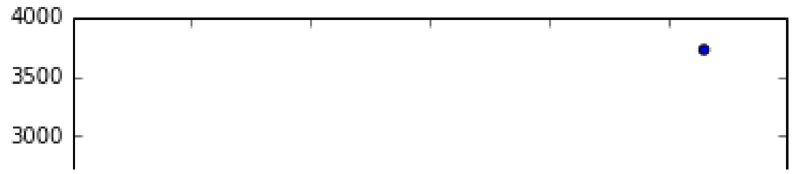
Out[24]:
```
<matplotlib.axes.AxesSubplot at 0x108df1690>
```

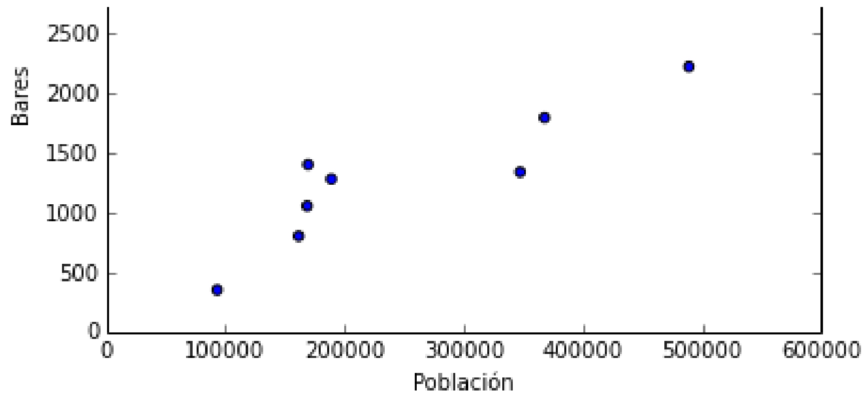


In [25]:
```
pd.tools.plotting.scatter_plot(joined, 'Población', 'Bares')
```

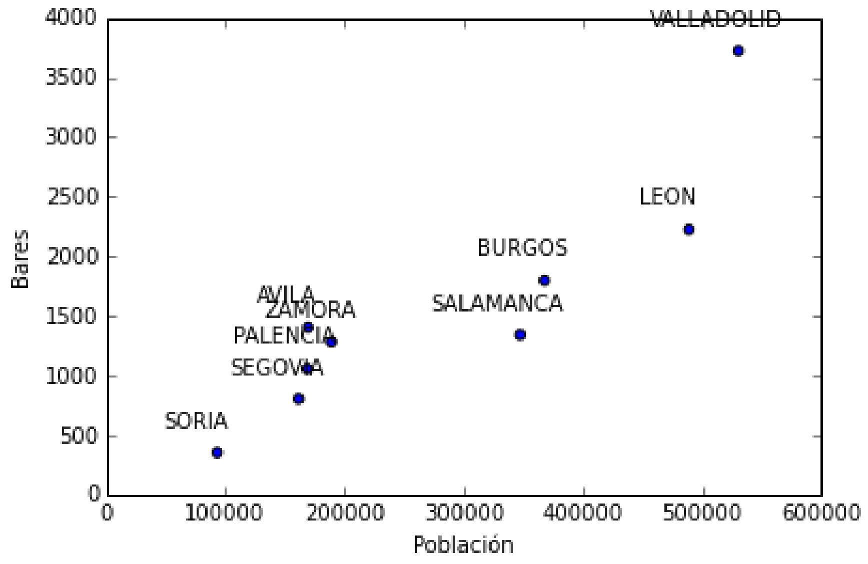
Out[25]:



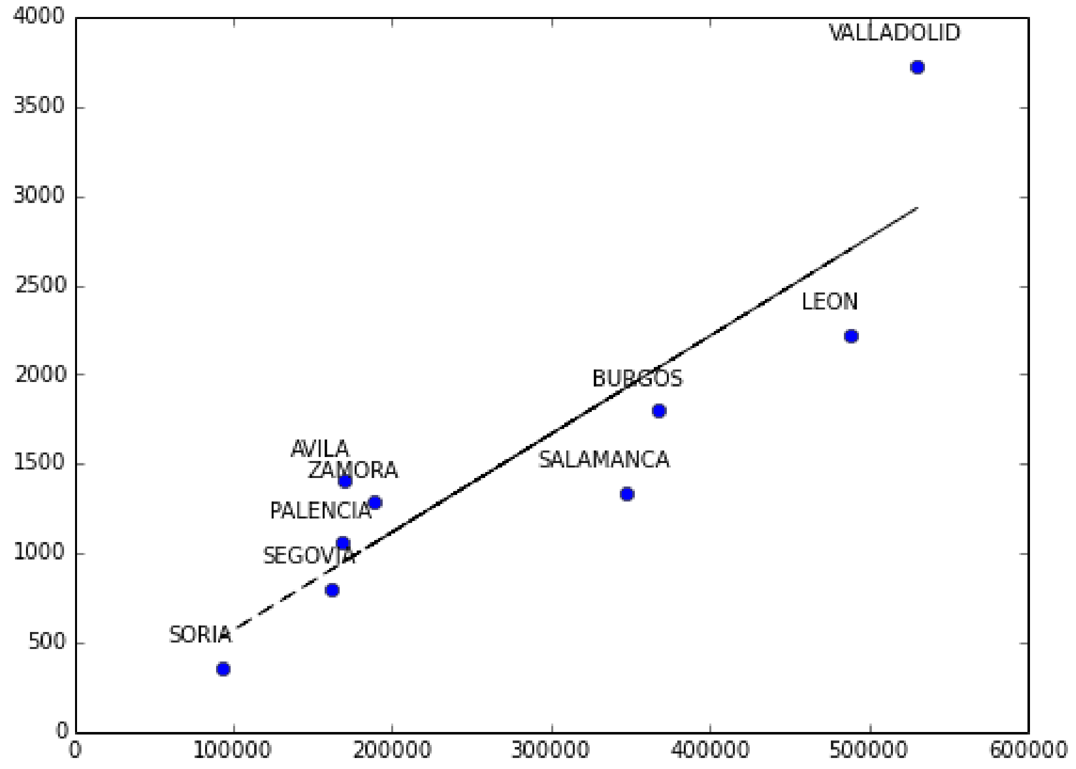




In [26]:
```
pd.tools.plotting.scatter_plot(joined, 'Población', 'Bares')

for label, x, y in zip(joined.index, joined['Población'], joined['Bares']
):
    plt.annotate(
        label,
        xy = (x, y), xytext = (-10, 10),
        textcoords = 'offset points', ha = 'center', va = 'bottom')
        #bbox = dict(boxstyle = 'round,pad=0.5', fc = 'yellow', alpha = 0
.2))
```



In [27]:
```
def scatter_and_fit(joined, figsize=(8, 6)):
    from pylab import polyfit, poly1d

    fit = polyfit(joined['Población'], joined['Bares'], 1)
    fit_fn = poly1d(fit)

    plt.figure(figsize=figsize, dpi=160)
    plt.plot(
        joined['Población'],
        joined['Bares'],
        'bo',
        joined['Población'],
        fit_fn(joined['Población']),
        '--k');


    for label, x, y in zip(joined.index, joined['Población'], joined['Bar
es']):
        plt.annotate(
            label,
            xy = (x, y), xytext = (-10, 10),
            textcoords = 'offset points', ha = 'center', va = 'bottom',)

scatter_and_fit(joined)
```



In [46]:
```
initial_length = len(bars)

capitals = [to_ascii(provincia) for provincia in bars.Provincia.unique()]
bars_caps = bars[bars.Localidad.apply(lambda x: x in capitals)]

print initial_length - len(bars_caps), '/', initial_length, "bars not in
capitals discarded.",
print len(bars_caps), "bars left."
```

```
8268 / 13984 bars not in capitals discarded. 5716 bars left.
```

In [47]:
```
by_municipio_gb = bars_caps.groupby('Municipio')
by_municipio = pd.DataFrame({'Bares': by_municipio_gb.size()})
```

```
by_municipio.set_index(by_municipio.index.map(to_ascii), inplace=True)
by_municipio
```

Out[47]:

	Bares
BURGOS	756
LEON	728
PALENCIA	405
SALAMANCA	589
SEGOVIA	209
SORIA	138
VALLADOLID	2276
ZAMORA	380
AVILA	235

In [48]:
```
population_caps = pd.read_excel('poblacion_por_provincias.xls', 'capitale
s', header=0, index_col='Capital')
population_caps.set_index(population_caps.index.map(to_ascii), inplace=Tr
ue)
population_caps = population_caps.rename_axis({'Total': 'Población'})
population_caps
```

Out[48]:

	Población
AVILA	59008
BURGOS	178966
LEON	134305
PALENCIA	82651
SALAMANCA	155619
SEGOVIA	55220
SORIA	39528
VALLADOLID	317864
ZAMORA	65525

In [49]:
```
population_caps.reset_index()
```

Out[49]:

	index	Población

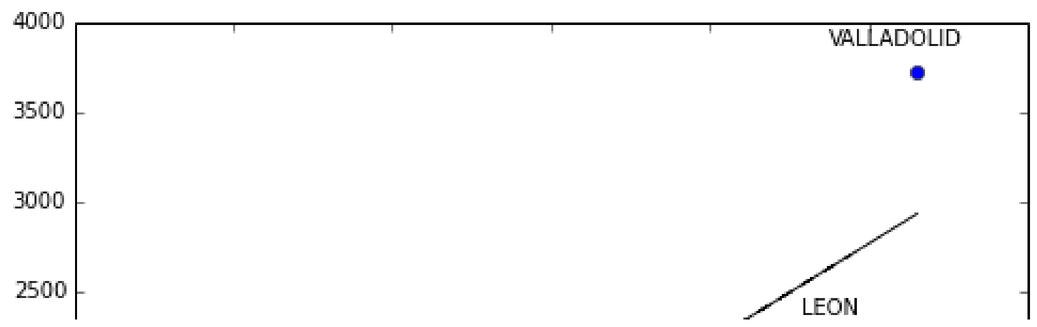
	index	Población
0	AVILA	59008
1	BURGOS	178966
2	LEON	134305
3	PALENCIA	82651
4	SALAMANCA	155619
5	SEGOVIA	55220
6	SORIA	39528
7	VALLADOLID	317864
8	ZAMORA	65525

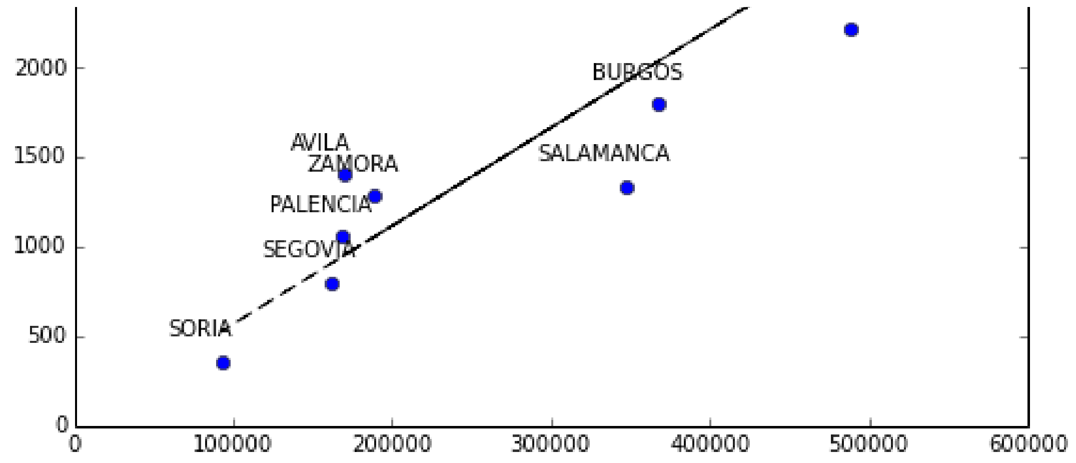
In [50]:
```
joined2 = pd.merge(by_municipio.reset_index(), population_caps.reset_inde
x()).set_index('index')
joined2
```

Out[50]:

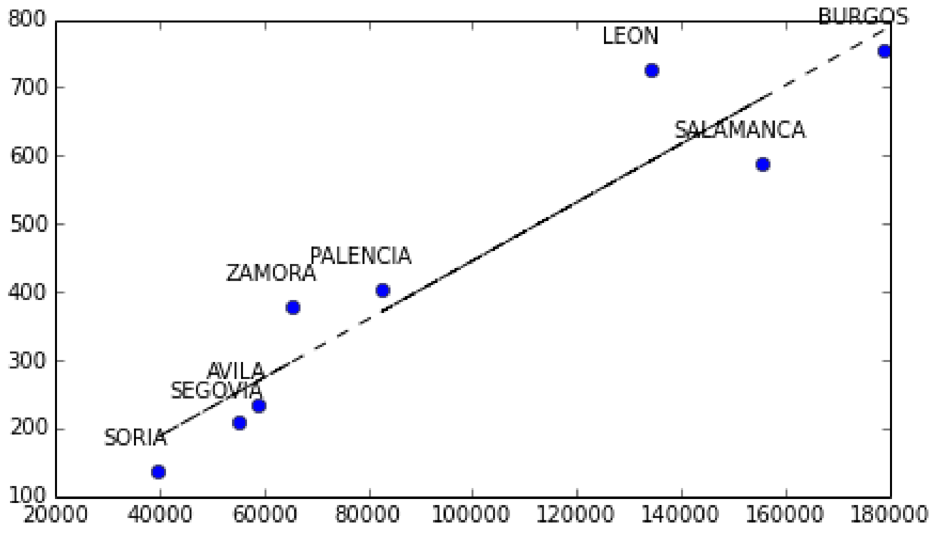
	Bares	Población
index		
BURGOS	756	178966
LEON	728	134305
PALENCIA	405	82651
SALAMANCA	589	155619
SEGOVIA	209	55220
SORIA	138	39528
VALLADOLID	2276	317864
ZAMORA	380	65525
AVILA	235	59008

In [51]:
```
scatter_and_fit(joined)
```

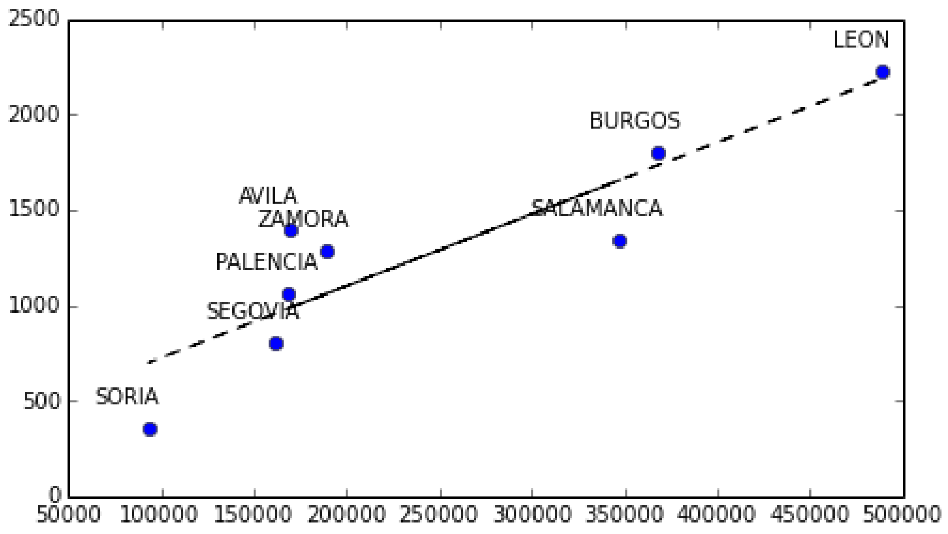




In [52]:
```
scatter_and_fit(joined2[joined2.index != 'VALLADOLID'], (7, 4))
```



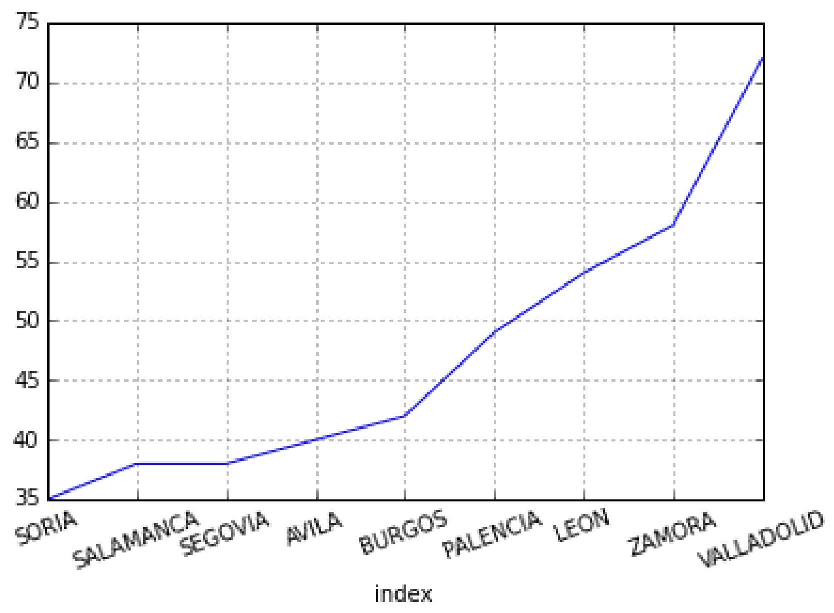
In [53]:
```
scatter_and_fit(joined[joined.index != 'VALLADOLID'], (7, 4))
```



In [54]:
```
joined2['Bares por 10K hab'] = 10**4 * joined2['Bares'] / joined2['Poblac
ión']
joined2['Bares por 10K hab'] = joined2['Bares por 10K hab'].apply(round)
```

```
joined2.sort('Bares por 10K hab')['Bares por 10K hab'].plot(rot=20)
```

Out[54]:
```
<matplotlib.axes.AxesSubplot at 0x10ae7f650>
```



In [55]:
```
localidades = bars.copy()
localidades['Localidad'] = localidades['Localidad'].apply(
    lambda x: to_ascii(x.split(' (')[0]).upper()) + ', ' + localidades['P
rovincia'].apply(lambda x: to_ascii(x).upper())
por_localidad = pd.DataFrame({'Bares': localidades.groupby('Localidad').s
ize()})
por_localidad[por_localidad.Bares >= 10]
por_localidad
```

Out[55]:
```
<class 'pandas.core.frame.DataFrame'>
Index: 2025 entries, ABADES, SEGOVIA to ZUZONES, BURGOS
Data columns (total 1 columns):
Bares     2025  non-null values
dtypes: int64(1)
```

In [56]:
```
por_localidad.head(10)
```

Out[56]:

	Bares
Localidad	
ABADES, SEGOVIA	4
ABEJAR, SORIA	1
ABEJERA, ZAMORA	1
ABELGAS DE LUNA, LEON	1
ABELON, ZAMORA	2

ABRAVESES DE TERA, ZAMORA	1
ABUSEJO, SALAMANCA	1
ACEBEDO, LEON	1
ACEBES DEL PARAMO, LEON	1
ADANERO, AVILA	4

In [57]:
```
# http://www.jcyl.es/web/jcyl/Estadistica/es/Plantilla100/1284253352941/_
/_/_

population_locs = pd.DataFrame()

for prov in joined.index:
    population_loc = pd.read_excel('poblacion_por_localidad.xls', prov, h
eader=0, index_col='Localidad')
    population_loc = population_loc.set_index(population_loc.index.map(
        lambda x: '{}, {}'.format(to_ascii(x).strip().split(', ')[0], pro
v)))
    population_loc = population_loc.rename_axis({'Total': 'Población'})
    print '{} + {}'.format(len(population_locs), len(population_loc))
    population_locs = pd.concat([population_locs, population_loc])
```
```
0 + 248
248 + 371
619 + 211
830 + 191
1021 + 362
1383 + 209
1592 + 183
1775 + 225
2000 + 248
```

In [58]:
```
population_locs.head()
```

Out[58]:

	Población
ADANERO, AVILA	266
ADRADA, AVILA	2704
ALBORNOS, AVILA	218
ALDEANUEVA DE SANTA CRUZ, AVILA	136
ALDEASECA, AVILA	273

In [59]:
```
print len(por_localidad), len(population_locs)
print sum([ix in por_localidad.index for ix in population_locs.index])
```
```
2025 2248
1382
```

In [60]:
```
df = por_localidad.join(population_locs, how='inner')
df
```

Out[60]:
```
<class 'pandas.core.frame.DataFrame'>
Index: 1382 entries, ADANERO, AVILA to ZAMORA, ZAMORA
Data columns (total 2 columns):
Bares         1382  non-null values
Población     1382  non-null values
dtypes: float64(1), int64(1)
```

In [61]:
```
df.columns
```

Out[61]:
```
Index([u'Bares', u'Población'], dtype=object)
```

In [62]:
```
df['Habitantes por bar'] = df[u'Población'] / df.Bares
df.head()
```

Out[62]:

	Bares	Población	Habitantes por bar
Localidad			
ADANERO, AVILA	4	266	66.50
ADRADA, AVILA	25	2704	108.16
ALBORNOS, AVILA	2	218	109.00
ALDEANUEVA DE SANTA CRUZ, AVILA	2	136	68.00
ALDEASECA, AVILA	1	273	273.00

In [63]:
```
df[df.Bares >= 50].sort('Habitantes por bar').head(10)
```

Out[63]:

	Bares	Población	Habitantes por bar
Localidad			
MEDINA DEL CAMPO, VALLADOLID	185	21594	116.724324
VALLADOLID, VALLADOLID	2276	311501	136.863357
BENAVENTE, ZAMORA	127	19259	151.645669
ZAMORA, ZAMORA	380	65362	172.005263

LEON, LEON	728	131680	180.879121
PALENCIA, PALENCIA	405	81198	200.488889
ARANDA DE DUERO, BURGOS	150	33459	223.060000
BURGOS, BURGOS	756	179906	237.970899
AVILA, AVILA	235	58915	250.702128
SALAMANCA, SALAMANCA	589	152048	258.146010

Back to top