15) Poisson Regression

Vitor Kamada

July 2018

Reference

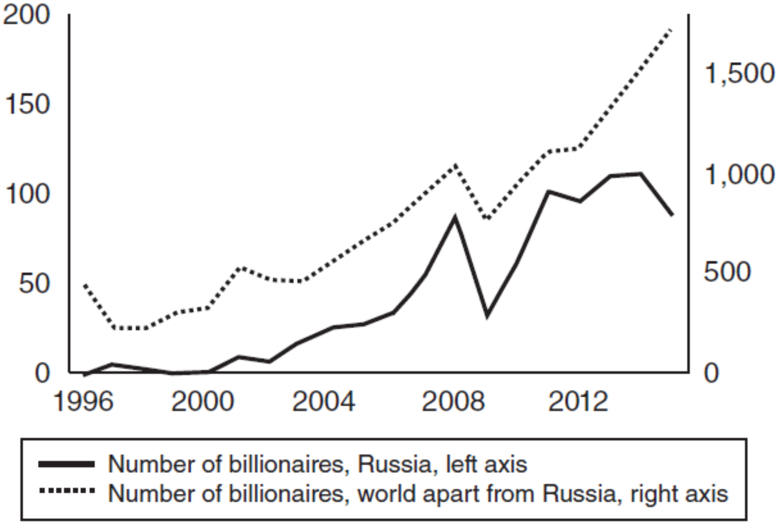
Tables, Graphics, and Figures from

Ch 4.4 Maximum Likelihood Estimation, by Sargent and Stachurski (2018)

https://lectures.quantecon.org/py/mle.html

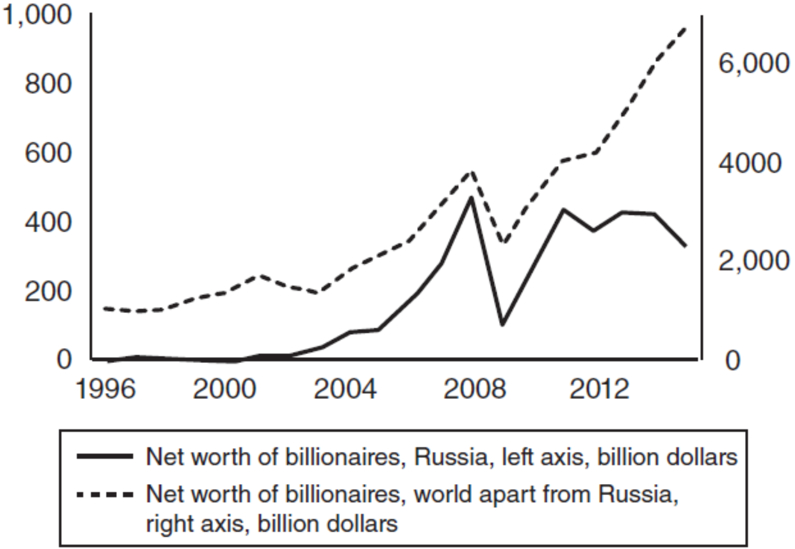
Russia's Billionaires, by Daniel Treisman (2016), in American Economic Review: Papers & Proceedings 2016, 106(5): 236–241

Number of Billionaires in Russia and the World



Source: Forbes

Net Worth of Billionaires in Russia and the World



Source: Forbes

Poisson Distribution

$$Pr(Y = y) = \frac{\mu^y}{y!} e^{-\mu}, \quad y = 0, 1, \ldots$$

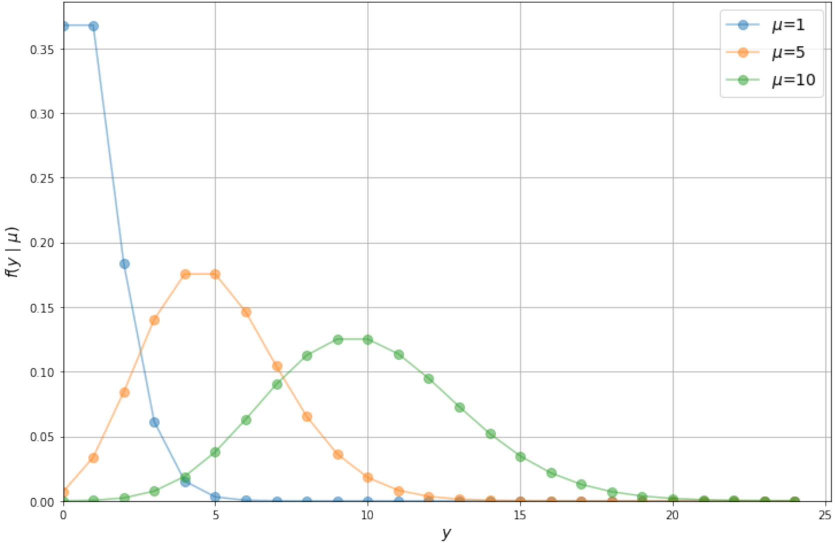
$$E(Y) = Var(Y) = \mu$$

```
from numpy import exp
from scipy.special import factorial
import matplotlib.pyplot as plt

poisson_pmf = lambda y, μ: μ**y / factorial(y) * exp(-μ)
y_values = range(0, 25)
```

Generate Graphic

```
fig, ax = plt.subplots(figsize=(12, 8))

for μ in [1, 5, 10]:
    distribution = []
    for y_i in y_values:
        distribution.append(poisson_pmf(y_i, μ))
    ax.plot(y_values,
            distribution,
            label=f'$\mu$={μ}',
            alpha=0.5,
            marker='o',
            markersize=8)

ax.grid()
ax.set_xlabel('$y$', fontsize=14)
ax.set_ylabel('$f(y \mid \mu)$', fontsize=14)
ax.axis(xmin=0, ymin=0)
ax.legend(fontsize=14)
```

plt.show()



Poisson MLE

$$Pr(Y = y) = \frac{e^{-\mu}\mu^{y}}{y!}$$

$$E[y|x] = \mu_i = exp(x_i'\beta)$$

$$lnL(\beta) = \sum_{i=1}^{N} \{y_i x_i'\beta - exp(x_i'\beta) - lny_i!\}$$

FOC: $$\sum_{i=1}^{N} (y_i - exp(x_i'\beta))x_i = 0$$

Interpretation of Regression Coefficients

$$E[y|x] = exp(x'\beta)$$

$$\frac{\partial E(y|x)}{\partial x_j} = \beta_j exp(x'\beta)$$

$$AME = \hat{\beta}_j \frac{1}{N} \sum_{i=1}^{N} exp(x_i'\hat{\beta})$$

If intercept is included, then $\hat{\beta}_j \bar{y}$

Forbes' Annual Rankings of Billionaires

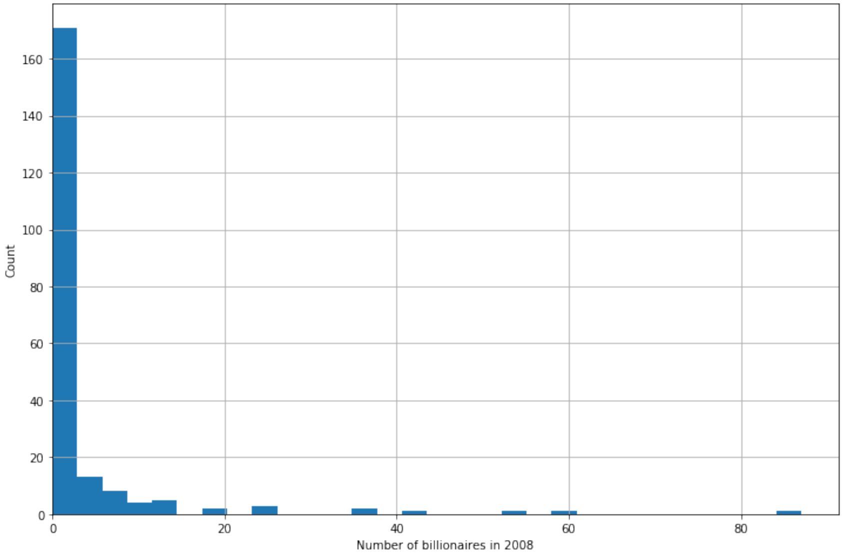
```python
import pandas as pd
pd.options.display.max_columns = 10

# Load in data and view
df = pd.read_stata('https://github.com/QuantEcon/QuantEcon.lectures.code/raw/master/mle/fp.dta')
df.head()
```

	country	ccode	year	cyear	numbil	...	topint08	rintr	noyrs	roflaw	nrrents
0	United States	2.0	1990.0	21990.0	NaN	...	39.799999	4.988405	20.0	1.61	NaN
1	United States	2.0	1991.0	21991.0	NaN	...	39.799999	4.988405	20.0	1.61	NaN
2	United States	2.0	1992.0	21992.0	NaN	...	39.799999	4.988405	20.0	1.61	NaN
3	United States	2.0	1993.0	21993.0	NaN	...	39.799999	4.988405	20.0	1.61	NaN
4	United States	2.0	1994.0	21994.0	NaN	...	39.799999	4.988405	20.0	1.61	NaN

of Billionaires per Country

```
numbil0_2008 = df[(df['year'] == 2008) & (
    df['country'] != 'United States')].loc[:, 'numbil0']

plt.subplots(figsize=(12, 8))
plt.hist(numbil0_2008, bins=30)
plt.xlim(xmin=0)
plt.grid()
plt.xlabel('Number of billionaires in 2008')
plt.ylabel('Count')
plt.show()
```

of Billionaires in 2008



Explaining the Number of Billionaires in 2008

	(1)	(2)	(3)	(4)
ln GDP per capita	1.08***	0.72***	0.74***	0.96***
	(0.14)	(0.25)	(0.23)	(0.24)
ln population	1.17***	0.81***	0.93***	1.15***
	(0.098)	(0.21)	(0.20)	(0.29)
Years in GATT/WTO	0.006	0.007	0.004	0.000
	(0.007)	(0.006)	(0.006)	(0.004)
ln market capitalization		0.40**	0.29*	0.11
		(0.17)	(0.17)	(0.24)
Real (lending) interest rate[a]		−0.010	−0.009	−0.007
		(0.00)	(0.010)	(0.009)
Top marginal income tax rate[b]		−0.051***	−0.058***	−0.060***
		(0.011)	(0.012)	(0.015)
Natural resource rents (percent GDP)			−0.005	0.013
			(0.011)	(0.013)
Rule of law, 2008			0.20	0.34
			(0.37)	(0.28)
Privatization proceeds, 1990–2008 (billion dollars)				−0.002*
				(0.001)
Observations	197	131	131	113
Pseudo-R^2	0.857	0.901	0.902	0.928
Countries with the largest under-predictions (number of billionaires in excess of model prediction):				
	USA (112)	Russia (42)	Russia (50)	Russia (33)
	Russia (60)	Germany (25)	Germany (22)	Germany (26)
	India (26)	USA (23)	India (16)	India (19)
	Hong Kong (24)	India (21)	USA (16)	Sweden (8)
	Turkey (23)	Hong Kong (11)	Hong Kong (11)	Lebanon (7)

Source: Treisman (2016)

Set Up Variables for Estimation

```python
df = df[df['year'] == 2008]

# Add a constant
df['const'] = 1

# Variable sets
reg1 = ['const', 'lngdppc', 'lnpop', 'gattwto08']
reg2 = ['const', 'lngdppc', 'lnpop',
        'gattwto08', 'lnmcap08', 'rintr', 'topint08']
```

import statsmodels.api as sm

```
poisson_reg = sm.Poisson(df[['numbil0']], df[reg1],
                         missing='drop').fit(cov_type='HC0')
print(poisson_reg.summary())
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 2.226090
         Iterations: 35
                          Poisson Regression Results
==============================================================================
Dep. Variable:                numbil0   No. Observations:                  197
Model:                        Poisson   Df Residuals:                      193
Method:                           MLE   Df Model:                            3
Date:                Wed, 26 Jul 2017   Pseudo R-squ.:                  0.8574
Time:                        15:41:38   Log-Likelihood:                -438.54
converged:                      False   LL-Null:                       -3074.7
                                        LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        -29.0495      2.578    -11.268      0.000     -34.103     -23.997
lngdppc        1.0839      0.138      7.834      0.000       0.813       1.355
lnpop          1.1714      0.097     12.024      0.000       0.980       1.362
gattwto08      0.0060      0.007      0.868      0.386      -0.008       0.019
==============================================================================
```

from statsmodels.iolib.summary2 import summary_col

```
regs = [reg1, reg2]
reg_names = ['Model 1', 'Model 2']
info_dict = {'Pseudo R-squared': lambda x: f"{x.prsquared:.2f}",
             'No. observations': lambda x: f"{int(x.nobs):d}"}
regressor_order = ['const',
                   'lngdppc',
                   'lnpop',
                   'gattwto08',
                   'lnmcap08',
                   'rintr',
                   'topint08',]
results = []

for reg in regs:
    result = sm.Poisson(df[['numbil0']], df[reg],
                        missing='drop').fit(cov_type='HC0', maxiter=100, disp=0)
    results.append(result)

results_table = summary_col(results=results,
                            float_format='%0.3f',
                            stars=True,
                            model_names=reg_names,
                            info_dict=info_dict,
                            regressor_order=regressor_order)
results_table.add_title('Table 1 - Explaining the Number of Billionaires in 2008')
```

print(results_table)

Table 1 - Explaining the Number of Billionaires in 2008

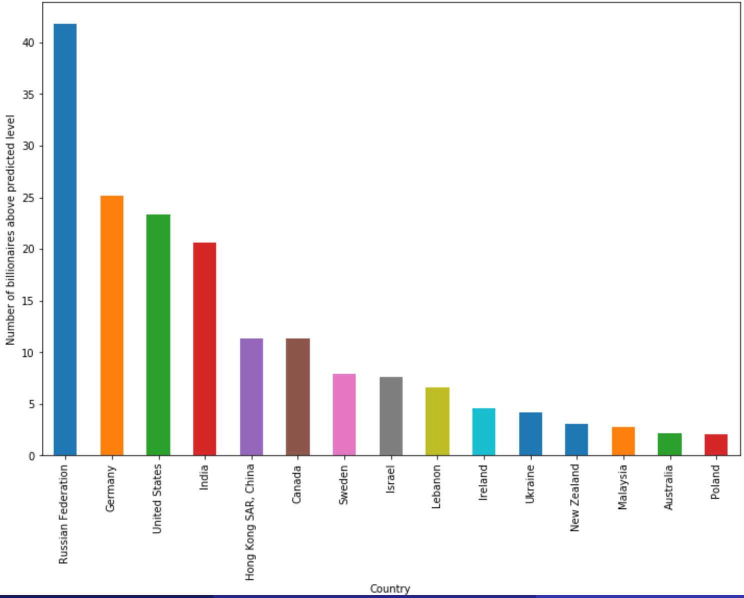
	Model 1	Model 2
const	-29.050***	-19.444***
	(2.578)	(4.820)
lngdppc	1.084***	0.717***
	(0.138)	(0.244)
lnpop	1.171***	0.806***
	(0.097)	(0.213)
gattwto08	0.006	0.007
	(0.007)	(0.006)
lnmcap08		0.399**
		(0.172)
rintr		-0.010
		(0.010)
topint08		-0.051***
		(0.011)
Pseudo R-squared	0.86	0.90
No. observations	197	131

Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01

Difference between the Predicted an Actual Values

```
data = ['const', 'lngdppc', 'lnpop', 'gattwto08', 'lnmcap08', 'rintr',
        'topint08', 'nrrents', 'roflaw', 'numbil0', 'country']
results_df = df[data].dropna()

# Use last model (model 3)
results_df['prediction'] = results[-1].predict()

# Calculate difference
results_df['difference'] = results_df['numbil0'] - results_df['prediction']

# Sort in descending order
results_df.sort_values('difference', ascending=False, inplace=True)

# Plot the first 15 data points
results_df[:15].plot('country', 'difference', kind='bar', figsize=(12,8), legend=False)
plt.ylabel('Number of billionaires above predicted level')
plt.xlabel('Country')
```

plt.show()



Number of billionaires above predicted level
0
5
10
15
20
25
30
35
40
Russian Federation
Germany
United States
India
Hong Kong SAR, China
Canada
Sweden
Israel
Lebanon
Ireland
Ukraine
New Zealand
Malaysia
Australia
Poland
Country

Percentage of Russian Billionaires in the Given Year who Made Their Wealth in the Following Sectors

	2005	2015
Oil, oil refining, gas, coal	56	28
Metals	48	20
Banking, finance, insurance	30	32
Telecom, IT, software, Internet	4	8
Construction	4	5
Real estate		19
Chemicals and fertilizer		13
Food and beverage production		7
Trade		6
Transport		3
Manufacturing		3
Casinos		1
Memo: Number of billionaires	*27*	*88*

Source: Forbes

Percentage of Those Becoming Billionaires in the Following Periods Who Were Still Billionaires in 2015

	All	USA	Germany	Japan	Brazil	Russia	Hong Kong
1996–2002	39	53	27	16	38	56	32
2003–2007	59	54	66	33	44	72	90
2008–2010	62	66	75	67	67	45	86
2011–2014	72	85	97	71	58	43	59

Source: Calculated from *Forbes*, various years.