CP.4.1.10, Sauer3

The following data, collected by the US Bureau of Economic Analysis, lists the year-over-year percent change in mean disposable personal income in the United States during 15 election years. Also, the proportion of the U.S. electorate that voted for the incumbent party's presidential candidate is listed. The first line of the table says that income increased by 1.49% from 1951 to 1952, and that 44.6% of the electorate voted for Adlai Stevenson, the incumbent Democratic party's candidate for president. Find the best least squares linear model for incumbent party vote as a function of income change. Plot this line along with the 15 data points. How many percentage points of vote can the incumbent party expect for each additional percent of change in personal income?

year	% income change	% incumbent vote
1952	1.49	44.6
1956	3.03	57.8
1960	0.57	49.9
1964	5.74	61.3
1968	3.51	49.6
1972	3.73	61.8
1976	2.98	49.0
1980	-0.18	44.7
1984	6.23	59.2
1988	3.38	53.9
1992	2.15	46.5
1996	2.10	54.7
2000	3.93	50.3
2004	2.47	51.2
2008	-0.41	45.7

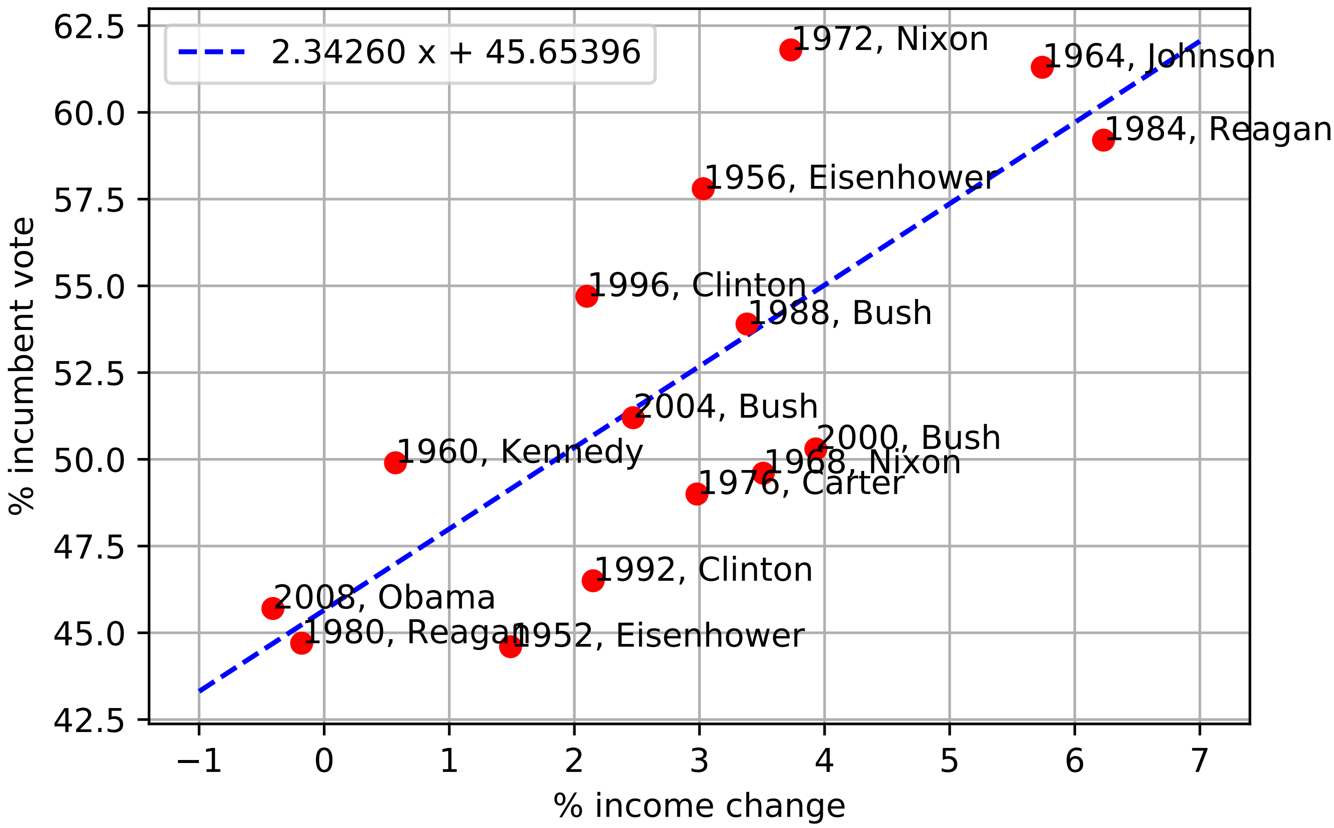
CP.4.1.10, Sauer3, solution, Langou

Colab: `https://colab.research.google.com/drive/1VOq1v8w5gNVNpgOGYQaZBKbY7l1PecQw`

The code (Python) can be found at the end of the document. The graph produced by the code is below. The x-axis represents the income change in percent. The y-axis represents the incumbent party vote in percent. A red cross represents a given year. We fit the data with a line and find that the best line in the least-squares sense is the line

$$y = 45.6540 + 2.3426\ x.$$

In other words, the incumbent party can expect 2.3426 percentage points of vote more for each additional percent of change in personal income.




```
import numpy as np
from math import sqrt
import matplotlib.pyplot as plt

data = np.array([
      [ 1952.,  1.49, 44.6 ],
      [ 1956.,  3.03, 57.8 ],
      [ 1960.,  0.57, 49.9 ],
      [ 1964.,  5.74, 61.3 ],
      [ 1968.,  3.51, 49.6 ],
      [ 1972.,  3.73, 61.8 ],
      [ 1976.,  2.98, 49.0 ],
      [ 1980., -0.18, 44.7 ],
      [ 1984.,  6.23, 59.2 ],
      [ 1988.,  3.38, 53.9 ],
      [ 1992.,  2.15, 46.5 ],
      [ 1996.,  2.10, 54.7 ],
      [ 2000.,  3.93, 50.3 ],
      [ 2004.,  2.47, 51.2 ],
      [ 2008., -0.41, 45.7 ] ]);

president_name = [
"Eisenhower",  # 1952
"Eisenhower",  # 1956
"Kennedy",     # 1960
"Johnson",     # 1964
```

```
"Nixon",        # 1968
"Nixon",        # 1972
"Carter",       # 1976
"Reagan",       # 1980
"Reagan",       # 1984
"Bush",         # 1988
"Clinton",      # 1992
"Clinton",      # 1996
"Bush",         # 2000
"Bush",         # 2004
"Obama"         # 2008
]

xx = data[:,1]
yy = data[:,2]

A = np.array([ np.ones( xx.shape ), xx ]).T
b = np.array([ yy ]).T
print("A =\n",A)
print("b =\n",b)

# compute x either using np.linalg.lstsq or normal equation methods
# x = np.linalg.lstsq(A,b,rcond=None)[0]
x = np.linalg.solve( A.T@A, A.T@b )

print("x =\n",x)
print("[ Ax, b ]=\n",np.concatenate((A@x,b),axis=1))
r = b - A@x
print("r =\n",r)
print("|| b - Ax ||_2 = ", f"{np.linalg.norm( r, 2 ):.4f}")
print("RMSE = ", f"{np.linalg.norm( r, 2 ) / sqrt( A.shape[0] ) :.4f}")
print("|| A^T ( b - Ax ) ||_oo =",
      f"{np.linalg.norm( A.T@( A@x - b), np.infty ):.1e}" )

xxx = np.linspace( -1., 7., 10)
yyy = x[0] + x[1] * xxx
plt.plot(xxx, yyy, '--b',label=f'{x[1,0]:7.5f}'+' x + '+f'{x[0,0]:7.5f}')
plt.plot( xx, yy, 'ro')
[ plt.text( xx[i], yy[i], f"{data[i,0]:.0f}"+", "+president_name[i]) for i in ran
plt.legend()
plt.xlabel('% income change')
plt.ylabel('% incumbent vote')
plt.grid()
plt.show()

# from google.colab import files
# plt.savefig('exercise_cp_4_1_10_________sauer3____plot.pdf');
# files.download('exercise_cp_4_1_10_________sauer3____plot.pdf')
```