PS2

April 30, 2018

1. 2D kernel density estimator

```
In [1]: import numpy as np
        bq_data = np.loadtxt('BQmat_orig.txt', delimiter=',')
```

```
In [5]: # (a)
        import matplotlib.pyplot as plt
        from mpl_toolkits.mplot3d import Axes3D
        %matplotlib notebook

        ages_vec = np.arange(18, 96)
        abils = np.array([0.25, 0.25, 0.20, 0.10, 0.10, 0.09, 0.01])
        abils_mdpts = np.array([0.125, 0.375, 0.60, 0.75, 0.85, 0.94, 0.995])
        abils_mat, ages_mat = np.meshgrid(abils_mdpts, ages_vec)

        fig = plt.figure()
        ax = fig.gca(projection='3d')
        ax.plot_surface(ages_mat, abils_mat, bq_data)
        ax.set_title('Distribution of bequest recipient proportion')
        ax.set_xlabel('Age')
        ax.set_ylabel('Lifetime Income Group')
        ax.set_zlabel('Percent of bequest received')
        plt.show()
```

```
<IPython.core.display.Javascript object>


<IPython.core.display.HTML object>
```

```
In [6]: # (b)
        from scipy.stats import gaussian_kde

        def get_scaled(bandwidth):
            prop_mat_inc = np.sum(bq_data, axis=0)
            prop_mat_age = np.sum(bq_data, axis=1)
            lrg_samp = 70000
```

```
        age_probs = np.random.multinomial(lrg_samp, prop_mat_age)
        income_probs = np.random.multinomial(lrg_samp, prop_mat_inc)
        age_freq = np.array([])
        inc_freq = np.array([])

        for age, num_s in zip(ages_vec, age_probs):
            vec_age_s = np.ones(num_s)
            vec_age_s *= age
            age_freq = np.append(age_freq, vec_age_s)

        for abil, num_j in zip(abils_mdpts, income_probs):
            vec_abil_j = np.ones(num_j)
            vec_abil_j *= abil
            inc_freq = np.append(inc_freq, vec_abil_j)

        data = np.vstack((age_freq, inc_freq))
        density = gaussian_kde(data, bw_method = bandwidth)

        coords = np.vstack([item.ravel() for item in [ages_mat, abils_mat]])
        BQkde = density(coords).reshape(ages_mat.shape)
        BQkde_scaled = BQkde / np.sum(BQkde)

        return BQkde_scaled
```

```
In [9]: def draw_scaled(data):
            fig = plt.figure()
            ax = fig.gca(projection='3d')
            ax.plot_surface(ages_mat, abils_mat, data)
            ax.set_title('Scaled distribution of bequest recipient proportion')
            ax.set_xlabel('Age')
            ax.set_ylabel('Lifetime Income Group')
            ax.set_zlabel('Scaled percent of bequest received')
            plt.show()
```

```
In [10]: for bandwidth in np.arange(0.05, 0.2, 0.05):
             draw_scaled(get_scaled(bandwidth))
```

```
<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>

<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

I will choose the bandwidth parameter as 0.1, as it reserves the unique pattern within each age group, and also it smooths the noise to some extent. The result is shown as below:

```
In [11]: draw_scaled(get_scaled(0.1))
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [12]: BQkde_scaled = get_scaled(0.1)
         print('The estimated density for bequest recipients who are age 61 in the 6th lifetim
         is', BQkde_scaled[43][6])
```

```
The estimated density for bequest recipients who are age 61 in the 6th lifetime income categor
```

2. Interaction terms

```
In [14]: import pandas as pd

         biden = pd.read_csv('biden.csv')
         biden.dropna(inplace=True)
         biden.head()
```

Out[14]:

	biden	female	age	educ	dem	rep
0	90.0	0	19.0	12.0	1.0	0.0
1	70.0	1	51.0	14.0	1.0	0.0
2	60.0	0	27.0	14.0	0.0	0.0
3	50.0	1	43.0	14.0	1.0	0.0
4	60.0	1	38.0	14.0	0.0	1.0

```
In [54]: from statsmodels.formula.api import ols
         model = ols(formula = "biden ~ age + educ + age * educ", data = biden)
         result = model.fit()
         print(result.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  biden   R-squared:                       0.018
Model:                            OLS   Adj. R-squared:                  0.016
Method:                 Least Squares   F-statistic:                     10.74
Date:                Mon, 30 Apr 2018   Prob (F-statistic):           5.37e-07
Time:                        08:34:37   Log-Likelihood:                -8249.3
No. Observations:                1807   AIC:                         1.651e+04
Df Residuals:                    1803   BIC:                         1.653e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     38.3735      9.564      4.012      0.000      19.617      57.130
age            0.6719      0.170      3.941      0.000       0.337       1.006
educ           1.6574      0.714      2.321      0.020       0.257       3.058
age:educ      -0.0480      0.013     -3.723      0.000      -0.073      -0.023
==============================================================================
Omnibus:                       64.246   Durbin-Watson:                   1.975
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               70.414
Skew:                          -0.481   Prob(JB):                     5.13e-16
Kurtosis:                       3.094   Cond. No.                     1.19e+04
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.19e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [16]:
```
result.cov_params()
```

Out[16]:

	Intercept	age	educ	age:educ
Intercept	91.461810	-1.545276	-6.725883	0.114416
age	-1.545276	0.029067	0.114149	-0.002159
educ	-6.725883	0.114149	0.509785	-0.008739
age:educ	0.114416	-0.002159	-0.008739	0.000166

Please find the coefficient parameters and standard error of the fitted model above.

In [17]:
```
b1 = 0.6719
b2 = 1.6574
b3 = -0.0480
var_b1 = 0.029067
var_b2 = 0.509785
var_b3 = 0.000166
cov_13 = -0.002159
```

```
cov_12 = 0.114149
cov_23 = -0.008739
```

(a)

$Y = \beta_0 + \beta_1 age + \beta_2 educ + \beta_3 age * educ$

The marginal effect of age on Joe Biden thermometer rating, conditional on education = $\beta_1 + \beta_3 educ$, and the standard error of the marignal effect = $\sqrt{(Var(\beta_1) + educ^2 * Var(\beta_3) + 2 * educ * Cov(\beta_1, \beta_3))}$

```
In [47]: marginal_age = pd.DataFrame(columns = ['educ', 'mar', 'std', 't'])
         marginal_age['educ'] = np.arange(0, 18)
         marginal_age['mar'] = b1 + marginal_age['educ'] * b3
         marginal_age['std'] = np.sqrt(var_b1 + marginal_age['educ']** 2 * var_b3 + 2 * margina
         marginal_age['t'] = marginal_age['mar'] / marginal_age['std']
```

```
In [48]: marginal_age
```

Out[48]:

	educ	mar	std	t
0	0	0.6719	0.170490	3.940983
1	1	0.6239	0.157845	3.952615
2	2	0.5759	0.145241	3.965129
3	3	0.5279	0.132691	3.978405
4	4	0.4799	0.120212	3.992104
5	5	0.4319	0.107829	4.005432
6	6	0.3839	0.095577	4.016649
7	7	0.3359	0.083516	4.021961
8	8	0.2879	0.071743	4.012958
9	9	0.2399	0.060424	3.970309
10	10	0.1919	0.049870	3.848018
11	11	0.1439	0.040682	3.537218
12	12	0.0959	0.033985	2.821809
13	13	0.0479	0.031417	1.524674
14	14	-0.0001	0.033926	-0.002948
15	15	-0.0481	0.040583	-1.185218
16	16	-0.0961	0.049749	-1.931683
17	17	-0.1441	0.060291	-2.390076

The magnitude of the marginal effect is decreasing with the increasing of education, and direction of the marginal effect changes from positive to negative. The statistical significance of marginal effect is pretty strong according to the t value we calculated.

(b)

$Y = \beta_0 + \beta_1 age + \beta_2 educ + \beta_3 age * educ$

The marginal effect of education on Joe Biden thermometer rating, conditional on age = $\beta_2 + \beta_3 age$, and the standard error of the marignal effect = $\sqrt{(Var(\beta_2) + age^2 * Var(\beta_3) + 2 * age * Cov(\beta_2, \beta_3))}$

```
In [49]: marginal_educ = pd.DataFrame(columns = ['age', 'mar', 'std', 't'])
         marginal_educ['age'] = np.arange(18, 94)
         marginal_educ['mar'] = b2 + marginal_educ['age'] * b3
         marginal_educ['std'] = np.sqrt(var_b2 + marginal_educ['age']** 2 * var_b3 + 2 * margin
         marginal_educ['t'] = marginal_educ['mar'] / marginal_educ['std']
```

```
In [50]: marginal_educ
```

```
Out[50]:     age     mar       std         t
         0    18  0.7934  0.498964  1.590095
         1    19  0.7454  0.487472  1.529113
         2    20  0.6974  0.476051  1.464968
         3    21  0.6494  0.464707  1.397438
         4    22  0.6014  0.453446  1.326289
         5    23  0.5534  0.442273  1.251265
         6    24  0.5054  0.431195  1.172092
         7    25  0.4574  0.420220  1.088477
         8    26  0.4094  0.409357  1.000106
         9    27  0.3614  0.398614  0.906642
         10   28  0.3134  0.388001  0.807729
         11   29  0.2654  0.377530  0.702990
         12   30  0.2174  0.367212  0.592028
         13   31  0.1694  0.357062  0.474428
         14   32  0.1214  0.347092  0.349763
         15   33  0.0734  0.337320  0.217597
         16   34  0.0254  0.327764  0.077495
         17   35 -0.0226  0.318442 -0.070971
         18   36 -0.0706  0.309375 -0.228202
         19   37 -0.1186  0.300588 -0.394560
         20   38 -0.1666  0.292104 -0.570344
         21   39 -0.2146  0.283952 -0.755760
         22   40 -0.2626  0.276161 -0.950894
         23   41 -0.3106  0.268762 -1.155669
         24   42 -0.3586  0.261788 -1.369810
         25   43 -0.4066  0.255274 -1.592796
         26   44 -0.4546  0.249257 -1.823821
         27   45 -0.5026  0.243772 -2.061759
         28   46 -0.5506  0.238858 -2.305138
         29   47 -0.5986  0.234549 -2.552137
         ..  ...     ...       ...       ...
         46   64 -1.4146  0.266700 -5.304083
         47   65 -1.4626  0.273980 -5.338347
         48   66 -1.5106  0.281661 -5.363182
         49   67 -1.5586  0.289712 -5.379827
         50   68 -1.6066  0.298102 -5.389424
         51   69 -1.6546  0.306804 -5.393011
         52   70 -1.7026  0.315793 -5.391512
         53   71 -1.7506  0.325043 -5.385748
```

```
54   72 -1.7986  0.334534 -5.376434
55   73 -1.8466  0.344246 -5.364194
56   74 -1.8946  0.354160 -5.349566
57   75 -1.9426  0.364260 -5.333011
58   76 -1.9906  0.374530 -5.314923
59   77 -2.0386  0.384958 -5.295637
60   78 -2.0866  0.395531 -5.275436
61   79 -2.1346  0.406238 -5.254560
62   80 -2.1826  0.417067 -5.233210
63   81 -2.2306  0.428011 -5.211554
64   82 -2.2786  0.439059 -5.189733
65   83 -2.3266  0.450206 -5.167862
66   84 -2.3746  0.461442 -5.146039
67   85 -2.4226  0.472763 -5.124342
68   86 -2.4706  0.484162 -5.102836
69   87 -2.5186  0.495634 -5.081573
70   88 -2.5666  0.507174 -5.060595
71   89 -2.6146  0.518776 -5.039936
72   90 -2.6626  0.530438 -5.019621
73   91 -2.7106  0.542156 -4.999669
74   92 -2.7586  0.553925 -4.980096
75   93 -2.8066  0.565743 -4.960911

[76 rows x 4 columns]
```

The magnitude of the marginal effect is first decreasing, then increasing with the increasing of age, and direction of the marginal effect changes from positive to negative. The statistical significance of marginal effect is pretty strong according to the t value we calculated, when age is larger than 30.

In [57]:
```
import statsmodels.api as sm
fig = plt.figure(figsize=(8,8))
fig = sm.graphics.plot_partregress_grid(result, fig=fig)
```

```
<IPython.core.display.Javascript object>


<IPython.core.display.HTML object>
```