Data Analysis with Python

(Class #6)

Gregory M. Eirich

QMSS

Agenda

1. Remaining OLS Diagnostics

2. Linear probability models

1. Remaining OLS Diagnostics

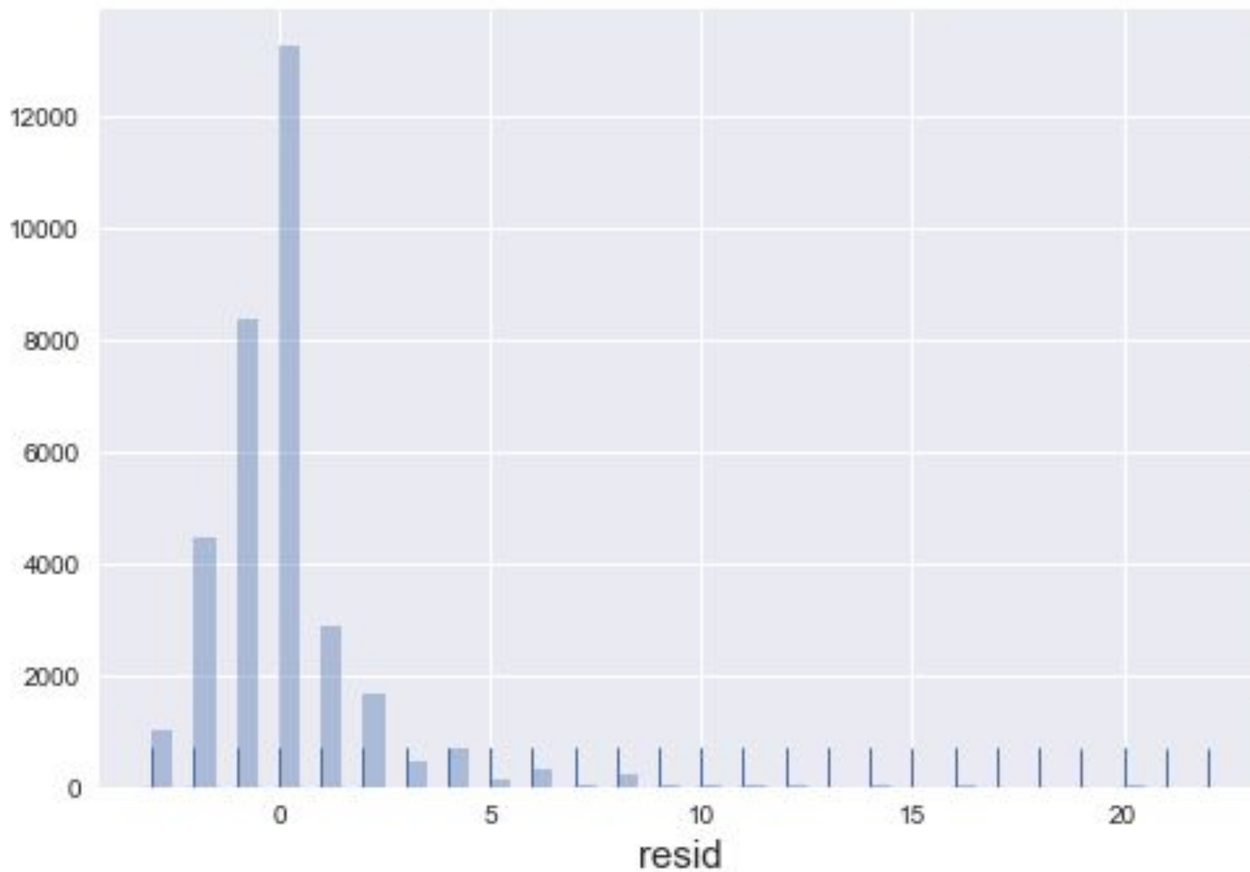
Assumption #6

Normality of the errors

- The errors should come from a (standard) normal distribution (N.B., errors ≠ residuals)

- Empirically, this is often not the case, but we can invoke the Central Limit Theorem and Law of Large Numbers to justify using the usual asymptotic inference, especially with reasonable sample sizes

The residuals from this regression are sort of normal, but not perfect



*

How I did that last graph

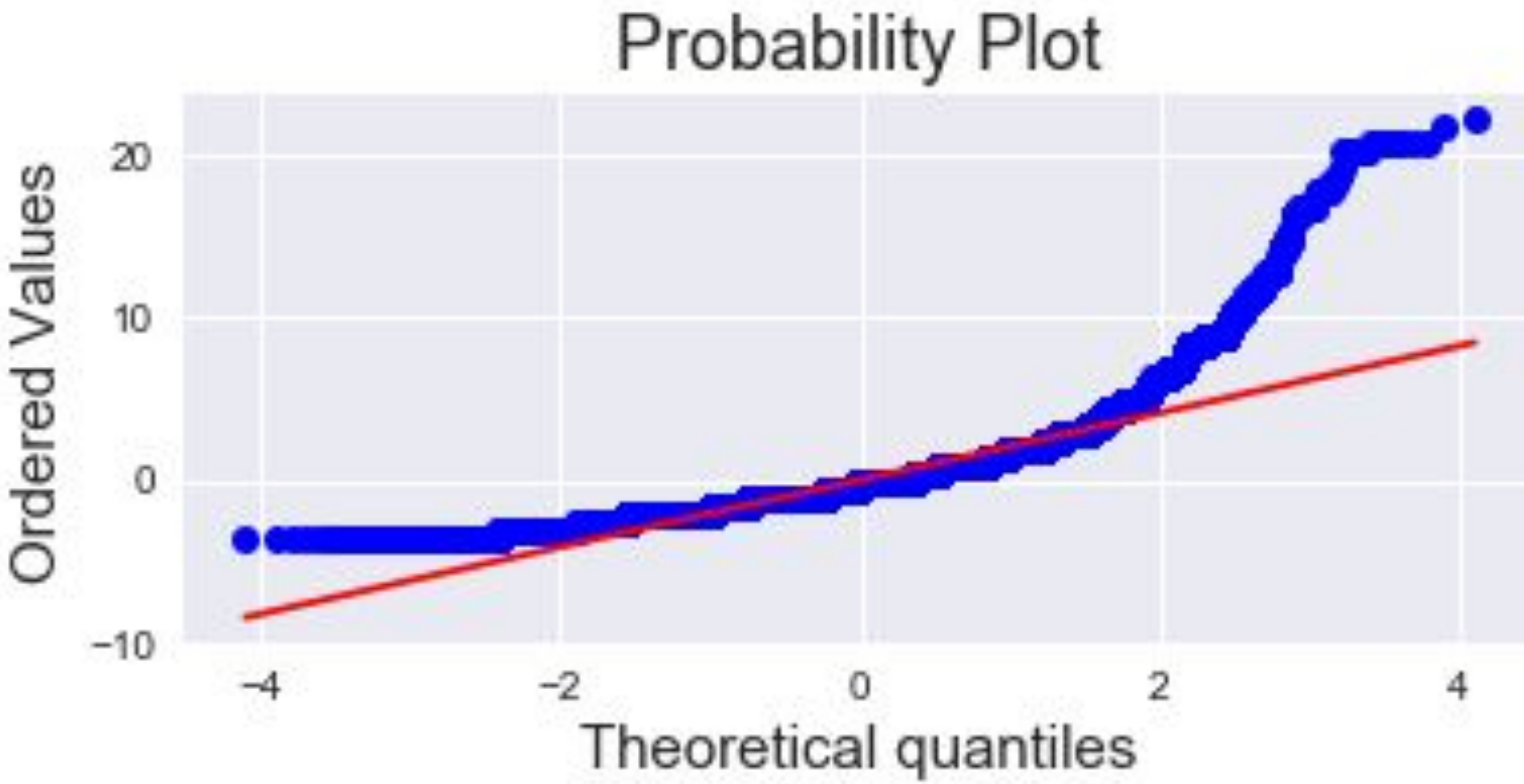
```
x = pd.to_numeric(x, errors='coerce')
##And for remove all rows with NaNs in column x use dropna:

x = x.dropna()
##Last convert values to ints:

x= x.astype(int)
##Make it integer to handle missings.

sns.distplot(x, kde=False, rug=True);
```

Normal Q-Q Plot



How I did that last graph

```
d['yhat'] = lm_tv.fittedvalues
d['resid'] = lm_tv.resid

dd = d.dropna(subset = ["resid"])

pred_val = lm_tv.fittedvalues.copy()
true_val = dd['tvhours'].values.copy()
residual = true_val - pred_val
fig, ax = plt.subplots(figsize=(6,2.5))
_ = ax.scatter(residual, pred_val)

fig, ax = plt.subplots(figsize=(6,2.5))
_, (__, ___, r) = scipy.stats.probplot(residual, plot=ax, fit=True)
r**2
```

Not an assumption, but …

No extreme outliers

- Certain observations should not throw off the whole regression line
- Outliers may be the result of data entry problems or real empirical differences among observations for some reason

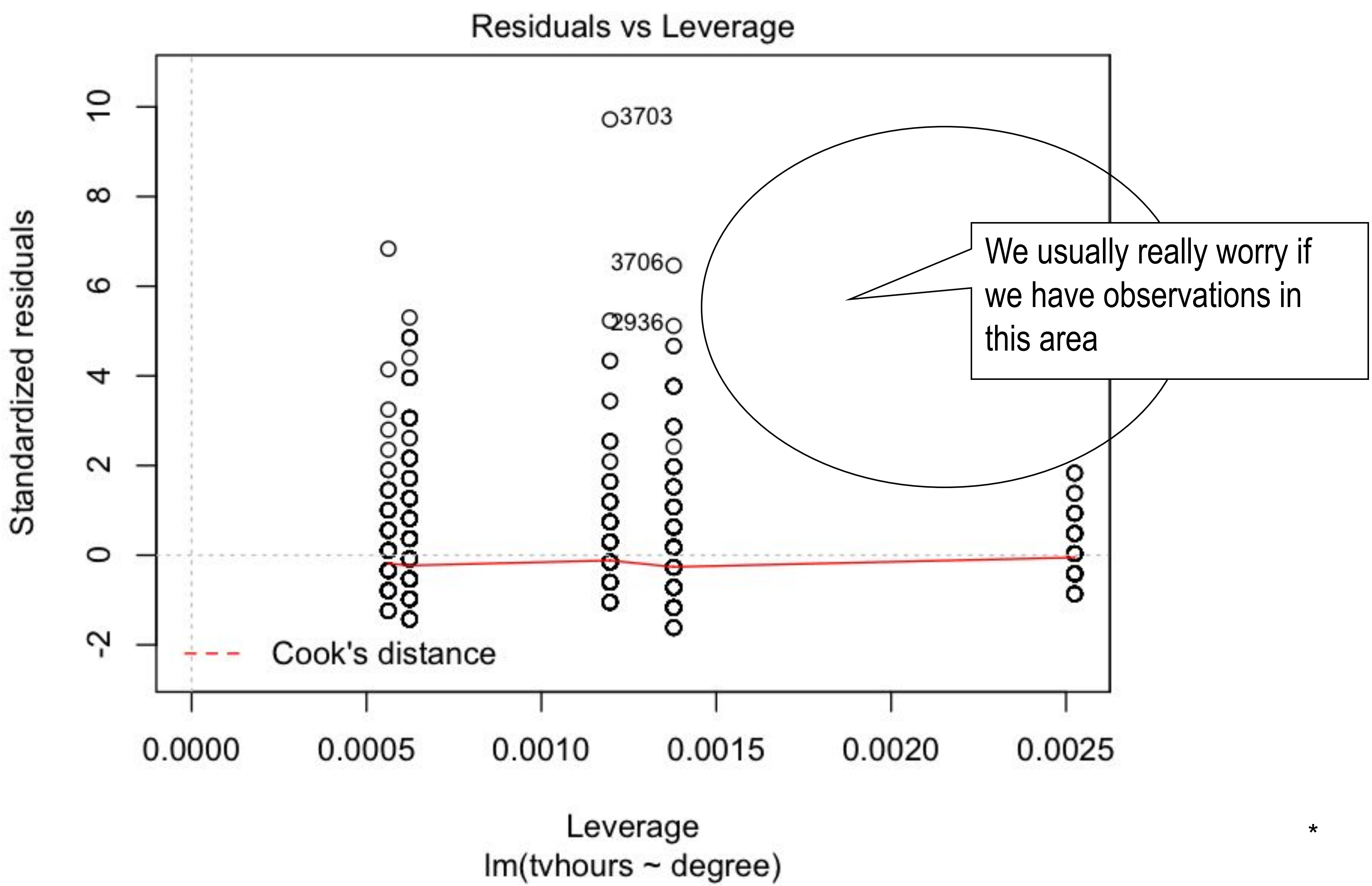
No Outliers

1. <u>Outliers</u> – extreme value on the residual, particularly worrisome on X

2. <u>Leverage</u> – big effect on the slope

3. <u>Influence</u> – big effect on other coefficients. Influence can be thought of as the product of leverage and outlierness.

Leverage vs. standardized residual plot



*

How I did that last graph

```
fig = sm.graphics.influence_plot(lm_tv)
```

How else can we look for influential cases?

- Cook's D = a measure of both the residual and leverage

Potential solutions to outliers...

Preliminary code

```
from __future__ import division
import pandas as pd
import numpy as np
import statsmodels.formula.api as smf
import os
from statsmodels.formula.api import ols, rlm
```

Robust regression

With greater weight given to well-behaved observations, we get a very similar and still statistically significant result

```
GSS = pd.read_csv("GSS_Cum.csv")
GSS_2010 = GSS[GSS.year == 2010] # subset on the year 2010

rlm_model = smf.rlm('tvhours ~ degree', GSS_2010).fit()
print(rlm_model.summary())
                    Robust linear Model Regression Results
==============================================================================
Dep. Variable:                tvhours   No. Observations:                 1426
Model:                            RLM   Df Residuals:                     1424
Method:                          IRLS   Df Model:                            1
Norm:                          HuberT
Scale Est.:                       mad
Cov Type:                          H1
Date:                Thu, 08 Jun 2017
Time:                        10:19:58
No. Iterations:                    15
==============================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
Intercept      3.2620      0.077     42.285      0.000         3.111     3.413
degree        -0.3864      0.038    -10.038      0.000        -0.462    -0.311          *
==============================================================================
```

Robust regression

- A complex algorithm is at work, involving calculating absolute deviations (not squared) and then also later performing weighted least squares (WLS)

```
Robust linear Model Regression Results
==============================================================================
Dep. Variable:                tvhours   No. Observations:                 1426
Model:                            RLM   Df Residuals:                     1424
Method:                          IRLS   Df Model:                            1
Norm:                          HuberT
Scale Est.:                       mad
Cov Type:                          H1
Date:                Thu, 08 Jun 2017
Time:                        10:19:58
No. Iterations:                    15
==============================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
Intercept      3.2620      0.077     42.285      0.000         3.111     3.413
degree        -0.3864      0.038    -10.038      0.000        -0.462    -0.311
==============================================================================
```

What was OLS again?

```
lm_tv = smf.ols(formula = "tvhours ~ degree", data = d).fit()
print (lm_tv.summary())

                            OLS Regression Results
==============================================================================
Dep. Variable:                tvhours   R-squared:                       0.054
Model:                            OLS   Adj. R-squared:                  0.054
Method:                 Least Squares   F-statistic:                     1934.
Date:                Mon, 03 Jun 2019   Prob (F-statistic):               0.00
Time:                        09:44:19   Log-Likelihood:                -75921.
No. Observations:               33788   AIC:                         1.518e+05
Df Residuals:                   33786   BIC:                         1.519e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      3.5967      0.019    190.058      0.000       3.560       3.634
degree        -0.4726      0.011    -43.977      0.000      -0.494      -0.452
==============================================================================
Omnibus:                19786.472   Durbin-Watson:                   1.920
Prob(Omnibus):              0.000   Jarque-Bera (JB):           269533.997
Skew:                       2.574   Prob(JB):                         0.00
Kurtosis:                  15.844   Cond. No.                         3.23
==============================================================================
```

Quantile (median) regression

For a 1 category increase in degree, the expected median of TV watching goes down by 0.50

```
quantreg_model = smf.quantreg('tvhours ~ degree', GSS_2010).fit()

print(quantreg_model.summary())

                         QuantReg Regression Results
==============================================================================
Dep. Variable:                tvhours   Pseudo R-squared:              0.02140
Model:                       QuantReg   Bandwidth:                      0.6510
Method:                 Least Squares   Sparsity:                        4.326
Date:                Thu, 08 Jun 2017   No. Observations:                 1426
Time:                        10:22:56   Df Residuals:                     1424
                                        Df Model:                            1
==============================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
Intercept      3.5000      0.094     37.400      0.000         3.316     3.684
degree        -0.5000      0.047    -10.708      0.000        -0.592    -0.408
==============================================================================
```

Quantile regression

We get the expected median of Y, given a 1 unit change in X, instead of the expected value (mean) of Y.

Instead of

$$E[y|x] = \alpha_0 + \alpha_1 x$$

QR gives, for each quantile

$$Q[y|x] = \alpha_0 + \alpha_1 x$$

Interval]

Quantile regression- Famous example

Journal of Economic Perspectives—Volume 15, Number 4—Fall 2001—Pages 143–156

Quantile Regression

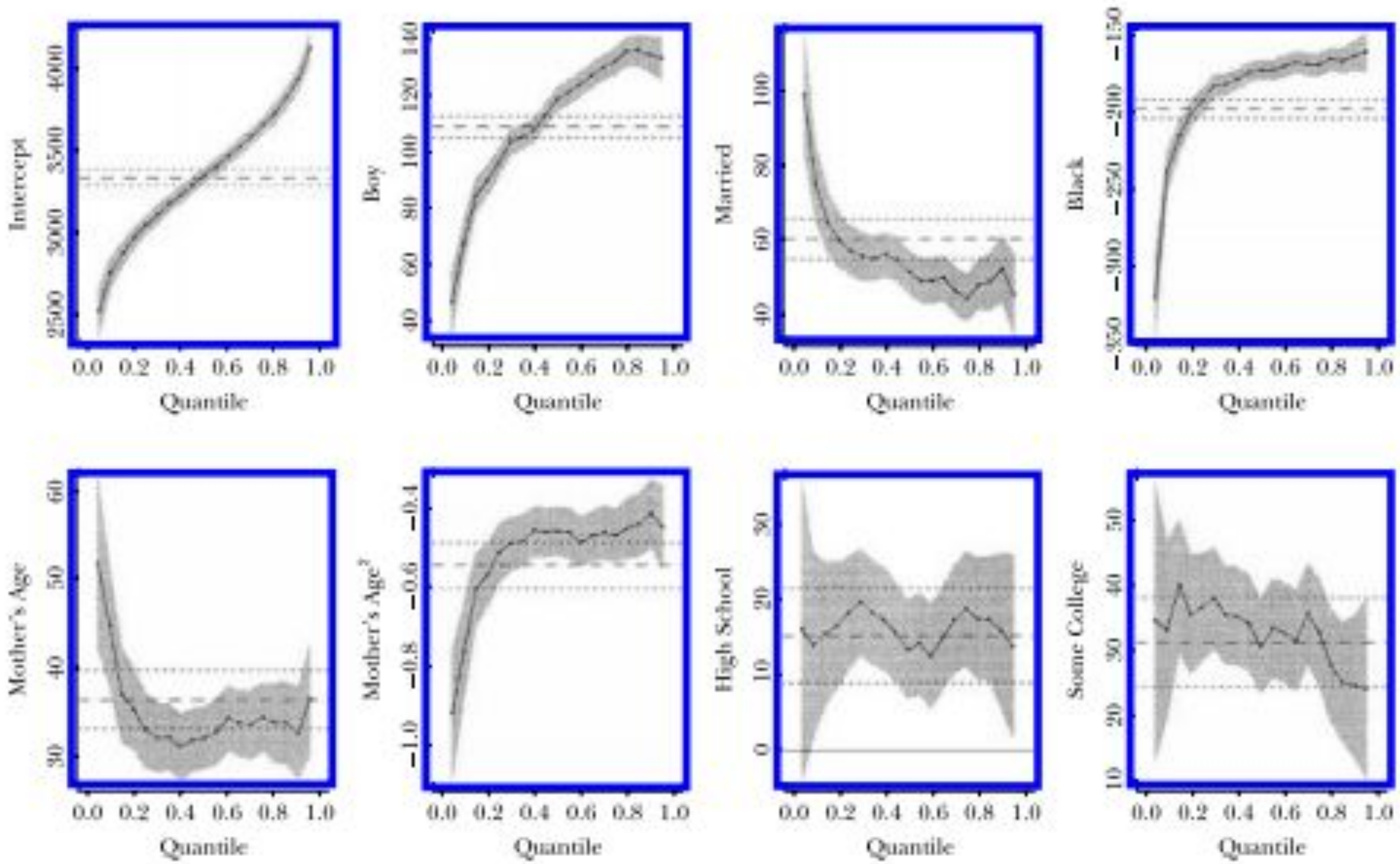
Roger Koenker and Kevin F. Hallock

We say that a student scores at the τth quantile of a standardized exam if

Quantile regression- Famous example

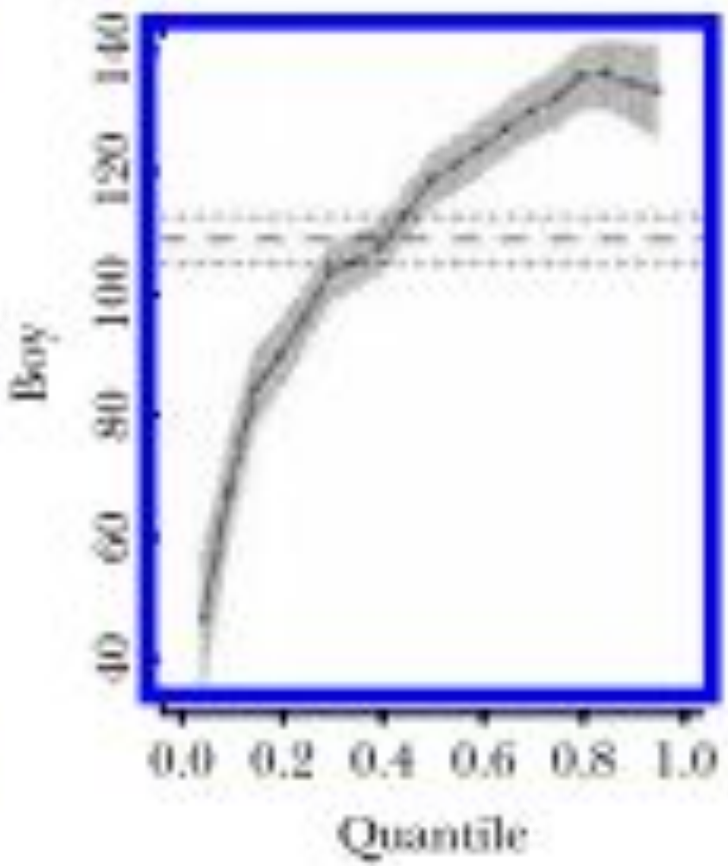
Figure 4

Ordinary Least Squares and Quantile Regression Estimates for Birthweight Model



Quantile regression- Famous example

"At any chosen quantile we can ask, for example, how different are the corresponding weights of boys and girls, given a specification of the other conditioning variables. The second panel answers this question. Boys are obviously larger than girls, by about 100 grams according to the ordinary least squares estimate of the mean effect, but as is clear from the quantile regression results, the disparity is much smaller in the lower quantiles of the distribution and considerably larger than 100 grams in the upper tail of the distribution. For example, boys are about 45 grams larger at the 0.05 quantile but are about 130 grams larger at the 0.95 quantile. The conventional least squares confidence interval does a poor job of representing this range of disparities."



The trade-off on all these tests, etc.

Interpretability *vs*. satisfaction of OLS assumptions

2. Linear probability model

Dummy Variables

With a dummy variable as the dependent variable, then this is called a Linear Probability Model

Later, we will replace this LPM with logistic and probit models, but for now, it is fine

A Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

Preliminary code:

```
from __future__ import division
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
import statsmodels.stats.api as sms
from statsmodels.compat import lzip
import os
import matplotlib.pyplot as plt
from patsy import dmatrices
```

Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

```
os.chdir("C:/Users/gme2101/Desktop/Data Analysis Data")
AdHealth = pd.read_csv("add-health.csv", low_memory=False)
AdHealth


AdHealth["relationship"] = AdHealth["H1RR1"]
AdHealth["attractive"] = AdHealth["H1IR1"]
AdHealth["smoking"] = AdHealth["H1TO1"]
AdHealth["birthYear"] = AdHealth["H1GI1Y"]
AdHealth["momEduc"] = AdHealth["H1NM4"]
AdHealth["noClubs"] = AdHealth["S44"]
```

Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

```
AdHealth["relationship"]
```

```
0       1
1       1
2       0
3       1
…

Name: relationship, Length: 6504, dtype: int64
```

Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

```
relationship_temp = pd.Categorical(AdHealth["relationship"], categories = [0, 1, 6, 8, 9], ordered = True)
 relationship_temp = pd.Categorical(AdHealth["relationship"], categories = [0, 1, 6, 8, 9], ordered = True)
 AdHealth["relationship"] = relationship_temp.rename_categories(["No", "Yes", "Refused", "Don't Know", "Not applicable"])
 AdHealth["relationship"]
```

```
Out[6]:
0        Yes
1        Yes
2         No
3        Yes
…

Name: relationship, Length: 6504, dtype: category
 Categories (5, object): [No < Yes < Refused < Don't Know < Not applicable]
```

Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

```
relationship_tab = pd.crosstab(index=AdHealth["relationship"], columns="count")
relationship_tab
relationship_tab/relationship_tab.sum()
```

col_0	count
relationship	
No	0.442497
Yes	0.550738
Refused	0.003383
Don't Know	0.003075
Not applicable	0.000308

Linear Probability Model

Among 13-17 olds, how many have had a romantic relationship?

```
# first replace "Refused", "Don't Know", and "Not applicable" entries with N/A values:
AdHealth.loc[AdHealth["relationship"] == "Refused","relationship"] = np.nan
AdHealth.loc[AdHealth["relationship"] == "Don't Know","relationship"] = np.nan
AdHealth.loc[AdHealth["relationship"] == "Not applicable","relationship"] = np.nan


AdHealth.loc[AdHealth['relationship'] == "Yes", 'romance'] = 1
AdHealth.loc[AdHealth['relationship'] == "No", 'romance'] = 0

pd.crosstab(index=AdHealth["romance"], columns="count")
```

col_0	count
romance	
0.0	2878
1.0	3582

What are some factors that may affect the likelihood of having a romantic relationship for young people?

Reason #1 - Attractiveness

```
AdHealth.loc[AdHealth["attractive"] >= 6 ,"attractive"] = np.nan

attractiveness_table = pd.crosstab(index=AdHealth["attractive"], columns="count")

attractiveness_table['perc'] =
100*attractiveness_table/attractiveness_table.sum()

attractiveness_table['cum_perc'] =
100*attractiveness_table["count"].cumsum()/attractiveness_table["count"].sum()

attractiveness_table
```

col_0	count	perc	cum_perc
attractive			
1.0	123	1.894056	1.894056
2.0	290	4.465661	6.359717
3.0	2788	42.931937	49.291654
4.0	2298	35.386511	84.678164
5.0	995	15.321836	100.000000

Reason #2 – Smoking or not?

```
# recode "smoking" variable
smoking_temp = pd.Categorical(AdHealth["smoking"], categories = [0, 1, 6, 8,
9], ordered = True)
AdHealth["smoking"] = smoking_temp.rename_categories(["No", "Yes", "Refused",
"Don't Know", "Not applicable"])

smoking_table = pd.crosstab(index=AdHealth["smoking"], columns="count")
smoking_table['perc'] = 100*smoking_table/smoking_table.sum()
smoking_table['cum_perc'] =
100*smoking_table["count"].cumsum()/smoking_table["count"].sum()
smoking_table
```

col_0	count	perc	cum_perc
smoking			
No	2863	44.019065	44.019065
Yes	3586	55.135301	99.154367
Refused	33	0.507380	99.661747
Don't Know	21	0.322878	99.984625
Not applicable	1	0.015375	100.000000

Reason #2 – Smoking or not?

```
# first replace "Refused", "Don't Know", and "Not applicable" entries with
N/A values:
AdHealth.loc[AdHealth["smoking"] == "Refused","smoking"] = np.nan
AdHealth.loc[AdHealth["smoking"] == "Don't Know","smoking"] = np.nan
AdHealth.loc[AdHealth["smoking"] == "Not applicable","smoking"] = np.nan

AdHealth.loc[AdHealth['smoking'] == "Yes", 'smoking_2'] = 1
AdHealth.loc[AdHealth['smoking'] == "No", 'smoking_2'] = 0

smoking_table_2 = pd.crosstab(index=AdHealth["smoking_2"], columns="count")
smoking_table_2['perc'] = 100*smoking_table_2/smoking_table_2.sum()
smoking_table_2['cum_perc'] =
100*smoking_table_2["count"].cumsum()/smoking_table_2["count"].sum()
smoking_table_2
```

col_0	count	perc	cum_perc
smoking_2			
0.0	2863	44.39448	44.39448
1.0	3586	55.60552	100.00000

LPM: Romantic relationship

```
sub = AdHealth.dropna(subset = ['romance', 'attractive', 'smoking_2'])

lpm_romance = smf.ols(formula = "romance ~ attractive + smoking_2", data = sub).fit()
print (lpm_romance.summary())

           OLS Regression Results
==============================================================================
Dep. Variable:                romance   R-squared:                       0.066
Model:                            OLS   Adj. R-squared:                  0.066
Method:                 Least Squares   F-statistic:                     226.9
Date:                Tue, 18 Jun 2019   Prob (F-statistic):           6.13e-96
Time:                        10:38:00   Log-Likelihood:                -4398.8
No. Observations:                6418   AIC:                             8804.
Df Residuals:                    6415   BIC:                             8824.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.2307      0.026      8.794      0.000       0.179       0.282
attractive     0.0538      0.007      7.774      0.000       0.040       0.067
smoking_2      0.2374      0.012     19.661      0.000       0.214       0.261
==============================================================================
Omnibus:                    28873.898   Durbin-Watson:                   1.908
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              798.711
Skew:                          -0.208   Prob(JB):                    3.65e-174
Kurtosis:                       1.323   Cond. No.                         17.5
```

For every category increase in attractiveness, a young person is 5.4 percentage points*** more likely to have had a romantic relationship on average, net of smoking.

LPM: Romantic relationship

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.2307	0.026	8.794	0.000	0.179	0.282
attractive	0.0538	0.007	7.774	0.000	0.040	0.067
smoking_2	0.2374	0.012	19.661	0.000	0.214	0.261

Having smoked a cigarette (vs. not smoking one) makes a young person 24 percentage points*** more likely to have been in a relationship on average, controlling for attractiveness.