4) Model Misspecification

Vitor Kamada

July 2018

Reference

Tables, Graphics, and Figures from

https://www.quantopian.com/lectures

Lecture 17 Model Misspecification

Exclusion of Important Variables

```python
start = '2013-01-01'
end = '2015-01-01'
bench = get_pricing('SPY', fields='price', start_date=start, end_date=end)
a1 = get_pricing('LRCX', fields='price', start_date=start, end_date=end)
a2 = get_pricing('AAPL', fields='price', start_date=start, end_date=end)

# Perform linear regression and print R-squared values
slr12 = regression.linear_model.OLS(a2, sm.add_constant(a1)).fit()
slrb1 = regression.linear_model.OLS(a1, sm.add_constant(bench)).fit()
slrb2 = regression.linear_model.OLS(a2, sm.add_constant(bench)).fit()
print "R-squared values of linear regression"
print "LRCX and AAPL: ", slr12.rsquared
print "LRCX and SPY: ", slrb1.rsquared
print "AAPL and SPY: ", slrb2.rsquared
```

```
R-squared values of linear regression
LRCX and AAPL:  0.911422827778
LRCX and SPY:  0.874582528812
AAPL and SPY:  0.795923926958
```

Pull Pricing Data from Further Back

```
start = '2009-01-01'
end = '2015-01-01'
bench = get_pricing('SPY', fields='price', start_date=start, end_date=end)
a1 = get_pricing('LRCX', fields='price', start_date=start, end_date=end)
a2 = get_pricing('AAPL', fields='price', start_date=start, end_date=end)

# Perform linear regression and print R-squared values
slr12 = regression.linear_model.OLS(a2, sm.add_constant(a1)).fit()
slrb1 = regression.linear_model.OLS(a1, sm.add_constant(bench)).fit()
slrb2 = regression.linear_model.OLS(a2, sm.add_constant(bench)).fit()
print "R-squared values of linear regression"
print "LRCX and AAPL: ", slr12.rsquared
print "LRCX and SPY: ", slrb1.rsquared
print "AAPL and SPY: ", slrb2.rsquared
```

```
R-squared values of linear regression
LRCX and AAPL:  0.499823847226
LRCX and SPY:  0.747298792884
AAPL and SPY:  0.756319173866
```

Pricing Data for Five Different Assets

```
start = '2014-01-01'
end = '2015-01-01'
x1 = get_pricing('PEP', fields='price', start_date=start, end_date=end)
x2 = get_pricing('MCD', fields='price', start_date=start, end_date=end)
x3 = get_pricing('ATHN', fields='price', start_date=start, end_date=end)
x4 = get_pricing('DOW', fields='price', start_date=start, end_date=end)
y = get_pricing('PG', fields='price', start_date=start, end_date=end)

# Build a linear model using only x1 to explain y
slr = regression.linear_model.OLS(y, sm.add_constant(x1)).fit()
slr_prediction = slr.params[0] + slr.params[1]*x1

# Run multiple linear regression using x1, x2, x3, x4 to explain y
mlr = regression.linear_model.OLS(y,
            sm.add_constant(np.column_stack((x1,x2,x3,x4)))).fit()
mlr_prediction = mlr.params[0] + mlr.params[1]*x1 + \
       mlr.params[2]*x2 + mlr.params[3]*x3 + mlr.params[4]*x4
```

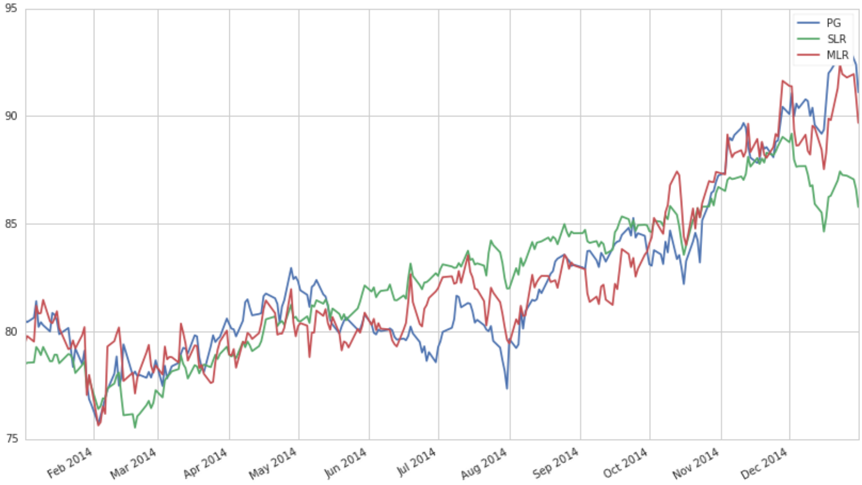
Adjusted R^2

```
print 'SLR R-squared:', slr.rsquared_adj
print 'MLR R-squared:', mlr.rsquared_adj

# Plot y along with the two different predictions
y.plot()
slr_prediction.plot()
mlr_prediction.plot()
plt.legend(['PG', 'SLR', 'MLR']);
```

```
SLR R-squared: 0.714538080242
MLR R-squared: 0.888347333447
```

SLR vs MLR Prediction



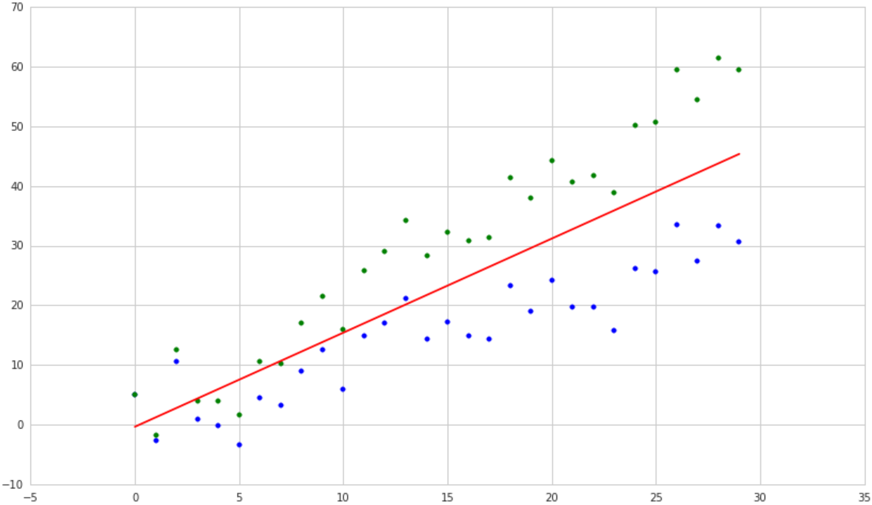
Pooling Different Populations

```
sample1 = np.arange(30) + 4*np.random.randn(30)
sample2 = sample1 + np.arange(30)
pool = np.hstack((sample1, sample2))

# Run a regression on the pooled data, with the
# independent variable being the original indices
model = regression.linear_model.OLS(pool,
    sm.add_constant(np.hstack((np.arange(30),
                    np.arange(30))))).fit()

# Plot the two samples along with the regression line
plt.scatter(np.arange(30), sample1, color='b')
plt.scatter(np.arange(30), sample2, color='g')
plt.plot(model.params[0] + \
         model.params[1]*np.arange(30), color='r')
```

Heteroskedasticity



Fundamentals Data

```
from quantopian.pipeline import Pipeline
from quantopian.research import run_pipeline
from quantopian.pipeline.data import Fundamentals
from quantopian.pipeline.experimental import QTradableStocksUS
```

```
import datetime
today = datetime.datetime.now().strftime('%Y-%m-%d')
today = '2017-01-1'
```

Create Pipeline Filters on the Market Cap

```
big_market_cap = Fundamentals.market_cap.latest > 1e8

universe =  QTradableStocksUS() &  big_market_cap

pipe = pipe = Pipeline(
    columns = {
     'free_cash_flow' : Fundamentals.free_cash_flow.latest,
     'operating_cash_flow' : Fundamentals.operating_cash_flow.latest,
     'industry' : Fundamentals.industry_template_code.latest,
      'total_revenue' : Fundamentals.total_revenue.latest,
    },
    screen = universe
)

data = run_pipeline(pipe, start_date = today, end_date = today)
```

data.head()

	free_cash_flow	industry	operating_cash_flow	total_revenue
Equity(2 [ARNC])	2.000000e+07	M	3.060000e+08	5.213000e+09
Equity(24 [AAPL])	1.208800e+10	N	1.612600e+10	4.685200e+10
Equity(31 [ABAX])	3.013000e+06	N	6.084000e+06	5.855200e+07
Equity(41 [ARCB])	1.322300e+07	T	3.547700e+07	7.139230e+08
Equity(52 [ABM])	-6.000000e+06	N	1.060000e+07	1.322300e+09

data = data[data.industry=='T']

data.head()

	free_cash_flow	industry	operating_cash_flow	total_revenue
Equity(41 [ARCB])	13223000.0	T	35477000.0	7.139230e+08
Equity(289 [MATX])	-32000000.0	T	8700000.0	5.004000e+08
Equity(300 [ALK])	138000000.0	T	307000000.0	1.566000e+09
Equity(1581 [CKH])	-76600000.0	T	13441000.0	2.069830e+08
Equity(1792 [CP])	NaN	T	NaN	NaN

Unscaled Model

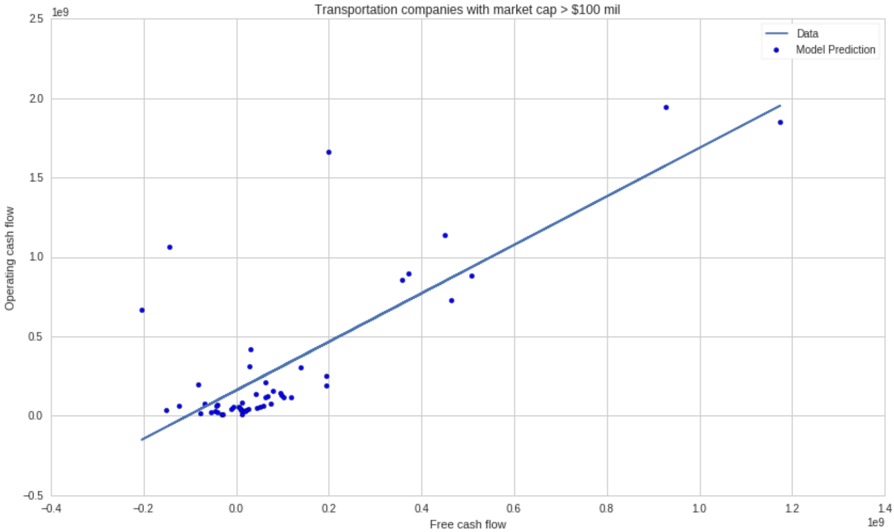
```
data.dropna(inplace=True)

unscaled_model = regression.linear_model.OLS(data['operating_cash_flow'],
                        sm.add_constant(data['free_cash_flow'])).fit()
prediction = unscaled_model.params[0] + \
             unscaled_model.params[1]*data['free_cash_flow']
print 'R-squared value of model:', unscaled_model.rsquared

# Plot the raw data for visualization
plt.scatter(data['free_cash_flow'], data['operating_cash_flow'])
plt.plot(data['free_cash_flow'], prediction)
plt.legend(['Data', 'Model Prediction'])
plt.xlabel('Free cash flow')
plt.ylabel('Operating cash flow')
plt.title('Transportation companies with market cap > $100 mil');
```

```
R-squared value of model: 0.612537429875
```

Transportation Firms with Market Cap > \$100 mil



Scaled Model

```
scaled_model = regression.linear_model.OLS(
    data['operating_cash_flow'].values/data['total_revenue'].values,
    sm.add_constant(data['free_cash_flow'].values/data['total_revenue'].values),
          missing='drop').fit()
print 'R-squared value of scaled model:', scaled_model.rsquared

prediction = scaled_model.params[0] + \
 scaled_model.params[1]*(data['free_cash_flow'].values/data['total_revenue'].values)
```

```
R-squared value of scaled model: 0.000555316621252
```

```
plt.scatter(data['free_cash_flow'].values/data['total_revenue'].values,
            data['operating_cash_flow'].values/data['total_revenue'].values)
plt.plot(data['free_cash_flow'].values/data['total_revenue'].values, prediction)
plt.legend(['Data', 'Model Prediction'])
plt.xlabel('Free cash flow/Total revenue')
plt.ylabel('Operating cash flow/Total revenue')
plt.title('Transportation companies with market cap > $100 mil');
```

Normalized by Revenue