Panel Data Models

Panel Data

Panels are a hybrid data structure that lives between the traditional data structures of microeconomics and forecasting.

- Contain observations of multiple individuals
  - Similar to standard cross-sectional data
- Contain multiple observations of each individual
  - Makes the data a collection of [possibly multivariate] time series data

Panel Data

Forecasting algorithms like ARIMA models, VAR models, and GAMs struggle to cope with this kind of data structure

- How do we difference out a time series when we have multiple observations (of different individuals) in any given period?
- How do we control for unobservable or unmeasurable differences between individuals?

Panel Data

Panel data allows us to generalize much of what we can learn through time series analysis

- We can generalize the effect of covariates to more than one individual
- We can make forecasts for different groups simultaneously from the same model
- BUT! We must have previous observations from all individuals in all periods (called a **balanced** panel)

Working with Panel Data

$$y_{it} = \alpha_{it} + X_{it}\beta + \epsilon_{it}$$

i: individual index, t: time index

We might start with the model above, but we wouldn't get far.

- We have insufficient information to calculate the model!
  - $K + NT > NT$

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \epsilon_{it}$$

If we remove the individual-level intercepts, we can remedy our information problem.

- Now, so long as we choose a reasonable number of covariates, $K < N$, our model is feasible

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \epsilon_{it}$$

Unfortunately, panel data means that we have correlated error terms within individuals.

- There is no good reason to believe

$$corr(y_{it}, y_{it+1}) = 0$$

- This is the same problem we saw with ARIMA models, but holds for each individual in our panel

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \epsilon_{it}$$

We need to decompose our error terms so that

$$\epsilon_{it} = \mu_i + \nu_{it}$$

where μ_i is an individual **fixed effect**, and ν_{it} is the noise term.

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \mu_i + \nu_{it}$$

Our model now has $K + N$ parameters, and NT degrees of freedom.

- So long as $K + N < NT$, we can now solve our model!

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \mu_i + \nu_{it}$$

The model can actually be solved using a modified form of OLS.

Working with Panel Data

$$y_{it} = \alpha + X_{it}\beta + \mu_i + \nu_{it}$$

$$\downarrow$$

$$y_{it} - \bar{y}_i = (X_{it} - \bar{X}_i)\beta + \nu_{it} - \bar{\nu}_i$$

$$\downarrow$$

$$\ddot{y}_{it} = \ddot{X}_{it}\beta + \ddot{\nu}_{it}$$

Working with Panel Data

$$\ddot{y}_{it} = \ddot{X}_{it}\beta + \ddot{\nu}_{it}$$

In effect, we difference each observation by subtracting the average values for a given individual over time, causing the intercept terms and individual fixed effects to be differenced out of the model.

$$\bar{X}_i = \frac{1}{T}\sum_{t=1}^{T} X_{it}$$

Robust Standard Errors

When we use panel data, we must consider that the variance in predictive power will vary by individual (some are more noisy than others)

- We can't just use standard OLS error functions
- Need to correct for the differences in variance between individuals

Robust Standard Errors

$$Var(\beta) = \sigma^2 (X'X)^{-1}(X'\Omega X)(X'X)^{-1}$$

but we can't know Ω. Instead, we need to estimate it.

1. Use OLS to estimate the model.
2. From OLS estimates, use the squared residuals to generate $\hat{\Sigma}$, an estimate of $\sigma^2\Omega$
3. Estimate $Var(\beta)$ as

$$(X'X)^{-1}(X'\hat{\Sigma}X)(X'X)^{-1}$$

4. In the case of clustered SE's, $\hat{\Sigma}$ is a blockwise diagonal matrix

Implementing A Fixed Effects Model

```
# Import Libraries
import pandas as pd
import numpy as np
import statsmodels.api as sm

# Import Data
# Put it back on one line!
data = pd.read_csv(
        'https://github.com/dustywhite7/Econ8310/raw
        /master/DataSets/firmInvestmentPanel.csv')

# Again, make the string one line...
y, x = pt.dmatrices("investment ~ market_value +
        capital + C(firm) + year + I(year**2)",
        data = data[data['year']<1954], return_type='dataframe')
```

Implementing A Fixed Effects Model

```
# Specify regression
reg = sm.OLS(endog=y, exog=x) # Last year saved for
                                   # forecast
# Fit regression with robust standard errors
fit = reg.fit().get_robustcov_results(cov_type='cluster',
             groups=data.loc[data['year']<1954,'firm'])
# Print results
print(fit.summary())
```

We can now explore our results, the effects of included variables, and what our forecasts might look like.

Implementing A Fixed Effects Model

```
# Store predictions and truth
xPred = pt.build_design_matrices([x.design_info],
        data.loc[data['year']>=1954, :])

pred = fit.predict(xPred).squeeze()
truth = data.loc[data.year>=1954, "investment"]
# Store errors
errors = pred - truth
# Calculate Absolute Percentage Error
pce = np.abs(errors/truth)*100
```

We need to determine how well we do at predicting out of sample with our current panel.

Implementing A Fixed Effects Model

```
# Print MSE, Mean Absolute Error,
#   and Mean Abs Percentage Error
print("Mean Squared Error: %s" %
        str(np.mean(errors**2)))
print("Mean Absolute Error: %s" %
        str(np.mean(np.abs(errors))))
print("Mean Absolute Percentage Error: %s"
        % str(np.mean(pce)))
```

```
Mean Squared Error: 13288.423957448418
```

```
Mean Absolute Error: 77.27884184438867
```

```
Mean Absolute Percentage Error: 58.25321343170577
```

In this case, it looks like we need more information…

Lab Time!