Patsy: Using Regression Equations

Why Use Patsy?

- We could just select our variables manually, and creating a column of ones is trivial
- Patsy allows us to separate our endogenous and exogenous variables AND to
  - "Dummy out" categorical variables
  - Easily transform variables (square, or log transforms, etc.)
  - Use identical transformations on future data

Getting Started

```
import patsy as pt
import pandas as pd
import numpy as np

data = pd.read_csv("wagePanelData.csv")

# To create y AND x matrices
y, x = pt.dmatrices("LWAGE ~ TIME + EXP + UNION + ED",
                    data = data)

# To create ONLY an x matrix
x = pt.dmatrix("~ TIME + EXP + UNION + ED",
                    data = data)
```

These regression equations automatically include an intercept term (Include `-1` as a variable to remove it)

Categorical Variables

```
# To create y AND x matrices
eqn = "LWAGE ~ C(ID) + TIME + EXP + UNION + ED + C(OCC)"
y, x = pt.dmatrices(eqn, data = data)
```

Categorical variables can be broken out into binary variables using the `C()` syntax inside of the regression equation.

In this case, there would be binary variables for each unique value of `ID` and `OCC`.

Interaction Terms

```
# To create y AND x matrices
eqn="LWAGE ~ C(ID) + EXP + C(UNION)*C(TIME) + ED + C(OCC)"
y, x = pt.dmatrices(eqn, data = data)
```

By using the `*` symbol, we can indicate to `patsy` that we want two (or more!) terms to be *interacted* with each other.

In the case above, this means that we will end up with a dummy variable for `UNION`, a dummy for each `TIME` value, and dummies for each period where `UNION` is also "True".

Transforming Variables

```
# To create y AND x matrices
eqn = "I(np.log(LWAGE)) ~ C(ID) + TIME + EXP + I(EXP**2)"
y, x = pt.dmatrices(eqn, data = data)
```

We can transform variables using the `I()` syntax inside of the regression equation. We then use any numeric transformation that we choose to impose on our data.

In this case we logged our dependent variable, `LWAGE`, and squared the `EXP` term.

Same Transformation on New Data!

```
# To create a new x matrix based on our previous version

xNew = pt.build_design_matrices([x.design_info], dataNew)
```

We can create a new matrix in the SAME SHAPE as our original `x` matrix by using the `build_design_matrices()` function in `patsy`.

We pass a list containing the old design matrix information, as well as the new data from which to construct our new matrix.

Why does Design Info Matter?

- Ensures that we always have the same number of categories
- Maintains consistency in our model
- Makes our work replicable

Using this method to create new datasets from which to generate predictions is extremely valuable