2. Regressions

Suppose we are interested in the connection between

- an outcome variable y (e.g. job satisfaction, engagement,...)
- and a variable x which may affect y (e.g. wage, the size of bonus payments, whether the firm uses performance pay or not,...)

Let e be a variable which describes all other determinants of y that we do not observe

Then we can denote the relationship between y and x as

$$y = f(x, e) \tag{1}$$

Key aim: Understand this function and learn about it by analyzing data

Distinction: Prediction and Causality

(i) Prediction

- Question: to what extent does knowing x allow us to *predict* y?
- Example:
  - When we as observers see that a company uses performance pay
  - What can we predict about the job satisfaction of its employees?
  - In other words: Is employee satisfaction higher in firms that use performance pay?

(ii) Causality

- Question: to what extent does a change of x *lead to* a change of y?
- Example:
  - A firm introduced performance pay
  - We want to know how this affected employee satisfaction
  - In other words: Did the change in performance pay *cause* a change in employee satisfaction?

These are different questions!

Further examples:

- *Education and wages*

  The fact that more educated people earn more does not tell us that education causes higher earnings

- *Gender diversity and performance*

  The fact that successful firms employ more women on boards does not tell us that a higher share of women causes a higher performance

Note:

- Answering the first (prediction) is typically substantially simpler than answering the second (causality)
- In the public debate (and also still in some fields in academia) these questions are often confounded
- We will start by thinking about the first question and then move to the second

The key idea of the following:

- Question: Why are regressions so important in empirical research?
- Answer:
  - Because they provide useful approximations to *conditional expectation functions*
  - And *conditional expectation functions* are a powerful tool to predict outcomes
- But:

  Without further ingredients they do not automatically detect causal relationships

2.1 The Conditional Expectation Function

- Think of X_i and Y_i as random variables (where X_i may be a vector)
- We are interested in the *conditional expectation function* (CEF) of Y_i given X_i in the population

$$E\left[Y_i \middle| X_i\right]$$

- Useful interpretation:

  Think of $E\left[Y_i \middle| X_i\right]$ as a function stating the mean of Y_i among all people who share the same value(s) of X_i

- If Y_i is discrete and takes values out of a set T

$$E[Y_i|X_i = x] = \sum_{t \in T} \Pr(Y_i = t|X_i = x) \cdot t$$

where $\Pr(Y_i = t|X_i = x)$ is the conditional probability that $Y_i = t$ when $X_i = x$

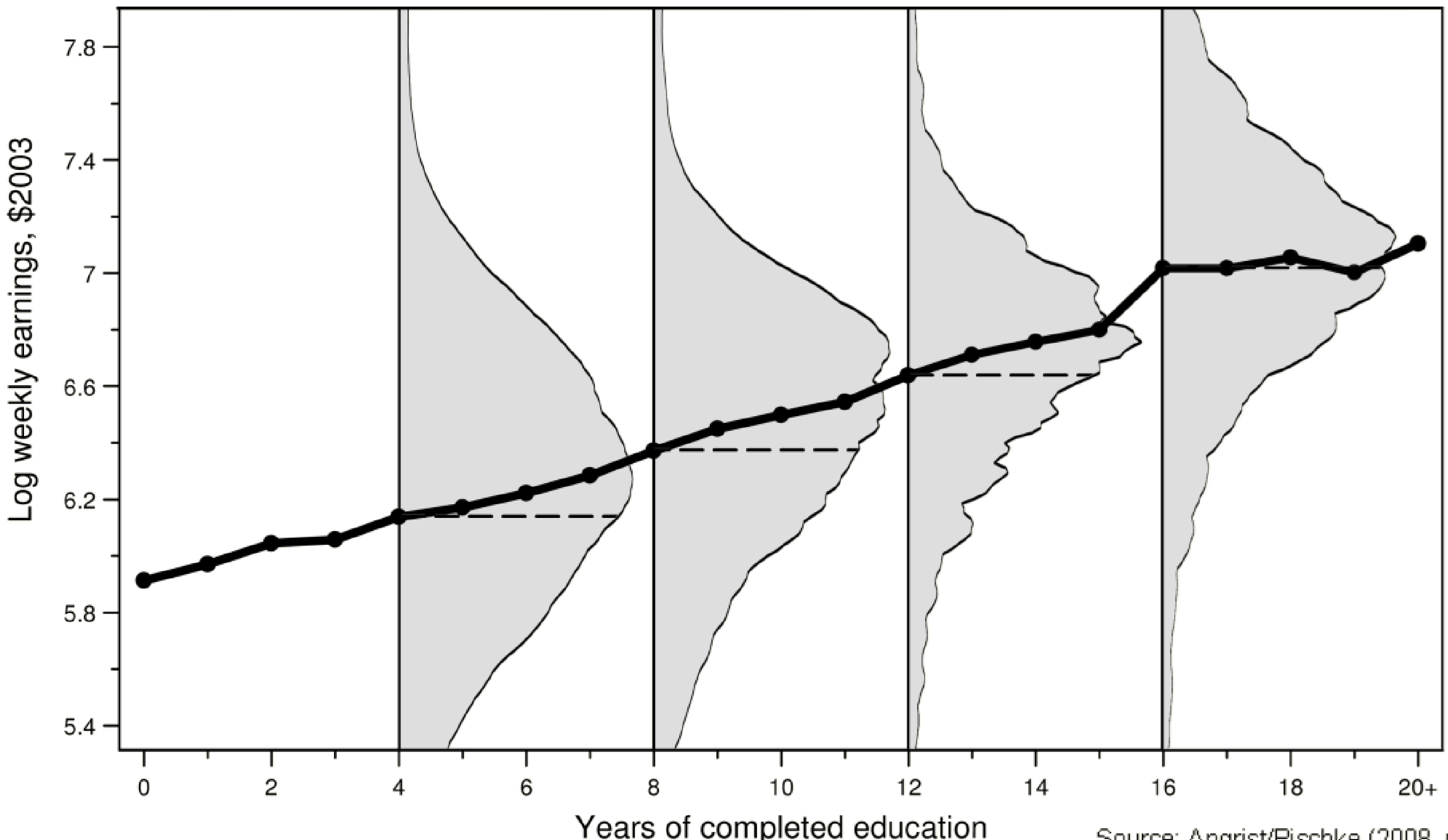
Distinguish:

- *Population:* Complete group of potential observations for our question (for example: all working age people living in Germany, all US firms...)

- A *sample*: the observations that we can use for our research
  - employees who take part in a survey study like the GSOEP or LPP
  - set of firms for which we have information on management practices
  - subjects taking part in an experiment

- We can estimate the population CEF from a representative sample
  - If we for instance observe pairs (Y_i, X_i) for $i = 1, \ldots, n$
  - We can estimate the conditional expectation of Y_i for a specific value of $X_i = x$ by taking the average of Y_i across observations with $X_i = x$

Example: The CEF of earnings as a function of years of education



Source: Angrist/Pischke (2008, p. 31)

Graphs in Python

- It is often useful to visualize data with graphs
- Particularly useful: package *Seaborn* (`import seaborn as sns`)

Examples:

- `sns.barplot(x='country', y='income', data=df)`
  - Plots one bar for each realization of x with height equal to mean of y
  - Note: illustrates the estimated CEF for categorial variables
  - Adds confidence bands: Estimates areas where true population mean lies with 95% probability
- `sns.relplot(x='income', y='happiness', data=df)`
  - Plots scatter plot where each dot is a data point
- `sns.distplot(df['wage'])`
  - Plots histogram of the variable
  - Note `df['x']` returns a series of all observations of variable x

Your Task

Feedback Talks and Employee Engagement

- Let us use the LPP to study the association between the use of feeback/appraisal interviews and employee engagement
- Please open again the Jupyter notebook `LPPanalysis.ipynb`
- Import further modules
  - `import statsmodels.api as sm`
  - `import statsmodels.formula.api as smf`
  - `import seaborn as sns`
- To estimate the CEF simply compare the mean of job engagement between employees who had an appraisal/feedback interview and those who didn't
  - Use the `enga_std` variable you generated before
  - `mmagespr` is a dummy variable which is equal to 1 if the employee had an appraisal interview and otherwise 0
  - Note: To do this, it is convenient to use the groupby method
    Syntax (adapt!): `df.groupby(df.country).wage.describe()`
- Visualize the CEF with a barplot
  (Adapt: `sns.barplot(x='country', y='income', data=df))`
- Save the notebook

Two key results (for the proofs see Angrist/Pischke (2009, pp 32)

Result: CEF Decomposition Property

We can decompose Y_i such that $Y_i = E\left[Y_i \middle| X_i\right] + \varepsilon_i$

(i) where ε_i is mean independent of X_i that is $E\left[\varepsilon_i \middle| X_i\right] = 0$

(ii) and therefore ε_i is uncorrelated with any function of X_i

- Therefore: A random variable Y_i can be decomposed into a piece that is "explained by X_i" (the Conditional Expectation Function) and a piece that remains unexplained by any function of X_i
- In the example: We can decompose the wage of a person
  - in a piece that is "explained" by education (i.e. the CEF)
  - and piece that is left over
  - and this latter piece is uncorrelated ("orthogonal to") with any function of education

Result: CEF Prediction Property

Let $m(X_i)$ be any function of X_i. The CEF solves

$$E[Y_i|X_i] = \arg\min_{m(X_i)} E\left[\left(Y_i - m(X_i)\right)^2\right]$$

so it is the best predictor of Y_i given X_i in the sense that it solves the minimum mean square error (MMSE) prediction problem.

- The CEF is a very useful predictor: If I observe other related variables and „plug them into the CEF“ the value of the CEF comes close to the true value of the outcome variable
- We want a function (call it $m(X_i)$) that gives us a good prediction for Y_i

$$\hat{Y}_i = m(X_i)$$

- Important criterion: The distance between $\hat{Y}_i$ and Y_i should be small
- The result now states: When we use the quadratic distance $\left(Y_i - m(X_i)\right)^2$, then the CEF is the best function we can find

Therefore:

- The CEF provides a natural summary of empirical relationships
  - It gives the population average of Y_i for the group of people having the same X_i
  - It describes the best (MMSE) predictor of Y_i given X_i
  - It allows to decompose Variance in the data (see Appendix)

- If I know the CEF I can make predictions which value Y_i would take for different values of X_i

  (Note: in the population; not in the sense of a causal change in Y_i because of a change of X_i!)

But: What is connection between the CEF and regression analysis and machine learning?

- In the following: regressions and other machine learning algorithms are tools to approximate the CEF

2.2 Regression and Conditional Expectations

- Typically, we will not know the functional form of the CEF when Y is a continuous variable
- But we can try to approximate it
- Start with simple case of two variables and consider the linear function

$$Y_i = \beta_0 + \beta_1 X_i$$

- Now determine β_0 and β_1 such that

$$(\beta_0, \beta_1) = \arg\min_{b_0, b_1} E\left[(Y_i - b_0 - b_1 X_i)^2\right]$$

- Let us call this the *Population Regression Function (PRF)*
- Of all possible linear functions of X_i – which one gives us the least (quadratic) deviation from Y_i in expected terms?

$$(\beta_0, \beta_1) = \underset{b_0, b_1}{\operatorname{argmin}} E[(Y_i - b_0 - b_1 X_i)^2]$$

First order conditions

$$E[2(Y_i - b_0 - b_1 X_i)] = 0$$
$$E[2(Y_i - b_0 - b_1 X_i)X_i] = 0$$

Hence,

$$b_0 = E[Y_i] - b_1 E[X_i]$$
$$b_1 E\left[X_i^2\right] = E[X_i Y_i] - b_0 E[X_i]$$

such that

$$b_1 = \frac{E[Y_i X_i]}{E\left[X_i^2\right]} - (E[Y_i] - b_1 E[X_i]) \frac{E[X_i]}{E\left[X_i^2\right]}$$

$$\Leftrightarrow b_1 = \frac{E[Y_i X_i] - E[Y_i]E[X_i]}{E\left[X_i^2\right] - (E[X_i])^2}$$

- Hence, in the bivariate case

$$\beta_1 = \frac{E[Y_i X_i] - E[Y_i]E[X_i]}{E\left[X_i^2\right] - (E[X_i])^2} = \frac{Cov[Y_i, X_i]}{V[X_i]}$$

- This is the population version of OLS regression for the bivariate case
- Define: The population residual

$$e_i = Y_i - b_0 - b_1 X_i$$

- Note that $Cov[e, X_i] = E[eX_i] - E[e]E[X_i] = 0$
  - as $E[Y_i - b_0 - b_1 X_i] = 0$
  - and $E[(Y_i - b_0 - b_1 X_i)X_i] = 0$ (from the first order conditions)
  - Hence, *the population residual is uncor*related with X_i
- Can proceed analogously in multivariate case (then β and X_i are vectors)

$$\beta = E[X_i X_i']^{-1} E[X_i Y_i]$$

From a Sample to the Population

- So far we spoke about whole populations but in reality we (typically) do not know the population parameters
- We work with samples (subsets) of a population but we want to say something about the population
- That is we want to estimate the population parameters β using a sample
- And we want to have an idea how good these estimates are

We want to

- obtain the estimated coefficients $\hat{\beta}$
- and learn about the precision of these estimates

The Bivariate Case: We want to estimate the parameter $\beta_1 = \frac{Cov[Y_i,X_i]}{V[X_i]}$

- We have a sample of size N and thus observe (Y_i, X_i) for $i = 1, \ldots N$
- We can estimate
  - $Cov[Y_i, X_i]$ by the sample covariance $\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})\,(Y_i - \bar{Y})$
  - $V[X_i]$ by the sample variance $\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})^2$
- And this leads to the OLS *estimator* $\hat{\beta} = \frac{\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y})}{\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})^2}$

Multivariate Case: We want to estimate $\beta = E[X_i X_i']^{-1} E[X_i Y_i]$

- We observe (Y_i, X_i') for $i = 1, \ldots N$, that is
  - $(Y_1, X_{10}, X_{11}, X_{12}, \ldots X_{1K-1})$,
  - $(Y_2, X_{20}, X_{21}, X_{22}, \ldots X_{2K-1})$,…
- We can estimate $E[X_i X_i']$ by $\frac{1}{N}\sum_{i=1}^{N} X_i\, X_i'$ and $E[X_i Y_i]$ by $\frac{1}{N}\sum_{i=1}^{N} X_i\, Y_i$
- And this leads to the OLS *estimator* $\hat{\beta} = \left[\sum_{i=1}^{N} X_i\, X_i'\right]^{-1} \sum_{i=1}^{N} X_i\, Y_i$

Python

OLS Regressions in Python

- We can use the module statsmodels & it is convenient to use “formulas”
- Import: `import statsmodels.formula.api as smf`
- If you have a DataFrame df containing variables y, x1 and x2 and you want to regress y (dependent variable) on x1 and x2 (indep. variables):
  ```
  reg = smf.ols('y ~ x1 + x2', data=df).fit()
  ```
- And show the results with
  ```
  print(reg.summary())
  ```
- Note: one can also directly get nice regression tables (as reported in research papers) with different specifications with `summary_col`
  ```
  from statsmodels.iolib.summary2 import summary_col
  ```
- Example:
  - `reg1 = smf.ols('y ~ x1', data=df).fit()`
  - `reg2 = smf.ols('y ~ x1 + x2', data=df).fit()`
  - `print(summary_col([reg1, reg2],stars=True))`

Python

Generating New Variables

- New variables can be created by `df['newvarname']=…`
- You can also generate new variables and compute their value as a function of existing variables:

```
df['salesPerEmp']=df['sales']/df['emp']
```

- A Boolean variable takes values *True* or *False*
  - A condition such as `(x>5)` gives back the value `True` when its true and otherwise `False`
- A Boolean variable can be used like a dummy variable, i.e. a variable which takes only value `0` or `1`
- A dummy variable can thus be created using a condition
  - Hence, `df['dummy']=(df['X']==5)` creates a dummy variable (column) that takes value `True` if the variable X is equal to 5

Study

Observational Data: Management Practices and Performance

Bloom und Van Reenen (2007), Bloom and Van Reenen (2012) study survey data

- Evaluate whether differences in the use management practices can explain productivity differences between firms
- Use an interview-based evaluation tool to assess 18 basic management practices
- Run the survey in many industries and countries
- Interviewers give a score from 1-5 on the 18 practices
- Compute a management score computed from the surveys
- Study the association between
  - the management score and
  - the financial success of the companies (e.g. sales, ROCE)

Management Practice Dimensions
(examples, see Bloom und Van Reenen (2010) , p. 206)

- Introduction of modern manufacturing techniques
- Rationale for introduction of modern manufacturing techniques
- Performance tracking
- Performance dialogue
- Consequence management
- Target time horizon
- Targets are stretching
- Managing human capital
- Promoting high performers
- Attracting human capital

Your Task | **Association between Management Practices & Performance**

- Use data from Bloom, Genakos, Sadun and Van Reenen. "Management Practices Across Firms and Countries." The Academy of Management Perspectives, 26, no. 1 (2012): 12-33.
- Start a new Juypiter Notebook (you can copy the first part with the imports and adapt from the previous exercise, but save it under a different name)
- Read the data into a DataFrame
  - `path_to_data = 'https://raw.githubusercontent.com/armoutihansen/EEMP2020/main/datasets/AMP_Data.csv'`
  - `df = pd.read_csv(path_to_data)`
- The data set for instance contains variables `management` (the management score across practices) and financial KPI `roce` (=EBIT/Capital employed)
- Type `df` to show the DataFrame
- Inspect the data set

Your Task — Association between Management Practices & Performance

- Inspect the data in more detail by plotting graphs, for instance use
  - `sns.displot(df.xvar)` to plot a histogram of a variable xvar
  - `sns.relplot(x='xvar', y='yvar', data=df)` for a scatter plot
  - `sns.regplot(x='xvar', y='yvar', data=df)` for a scatter plot that includes a regression line
- Now run a regression of `roce` as dependent variable on `management`
  - Recall the syntax (adapt!):
  - ```
    reg = smf.ols('yvar~xvar1+xvar2', data=df).fit()
    print(reg.summary())
    ```
- Interpret your result
- Save your notebook as `ManagementPractices` to reuse it later
    ```

2.3 Dummy Variables

When X_i is a single dummy variable that only takes value 0 or 1

- Then $E[Y_i|X_i = 0]$ is a constant and $E[Y_i|X_i = 1]$ is another constant and the CEF is fully characterized by these constants:

$$E\left[Y_i \middle| X_i\right] = \underbrace{E[Y_i|X_i = 0]}_{\beta_0} + X_i \cdot \underbrace{(E[Y_i|X_i = 1] - E[Y_i|X_i = 0])}_{\beta_1}$$

  is a linear function of X_i

- When I have precise estimates of the PRF then I have a precise estimate of $E\left[Y_i \middle| X_i\right]$

Note:

- The PRF exactly describes the CEF
- Linearity is not an assumption but a fact
- This is a very common data structure for instance in an experiment: X_i indicates whether somebody is in the treatment instead of the control group

Your Task — Regression & Conditional Expectation

- Please open again the Jupyter notebook `LPPanalysis.ipynb`
- Estimate a regression of (std.) engagement on the `mmagespr` dummy
- Compare the constant term (intercept) and coefficient of `mmagespr` with the conditional means computed in the last exercise. What do you see?
- Please inspect the robustness of the connection between engagement and the use of appraisal interviews
- To do so, estimate a multivariate regression adding the following further explanatory variables (variable names in parentheses):
  - Age (`alter`)
  - Manager (dummy `mleitung`)
  - Temporary contract (dummy `mbef`)
  - Part time work (dummy `maz_voll_teil`)
  - Works from home (dummy `mheim`)
  - Training (dummy `mwb`)

2.4 Interaction Terms

- Sometimes we expect that the conditional expectation function $E\left[Y_i \middle| X_{i1}, X_{i2}\right]$ is not additively separable such that it can sensibly be approximated by a population regression $Y_i = \alpha + \beta_1 X_{i1} + \beta_2 X_{i2}$
- But we may want to allow for the possibility that the effect of X_{i1} depends on the size of X_{i2}, for instance
  - The effect of performance pay on job satisfaction may depend on gender
  - The effect of a training may depend on experience,...
- In experiments we might consider a setting in which X_{i1} is a treatment dummy and X_{i2} is a specific characteristic of a treated object and we may want to study *heterogenous treatment effects*
- For instance the object is a
  - person and the characteristic is the age, gender, or experience.
  - firm and the characteristic is the size, industry, region,..

- When expecting that the effect of X_{i1} depends on the size of X_{i2} researchers typically estimate a regression

$$Y_i = \alpha + \beta_1 \cdot X_{i1} + \beta_2 \cdot X_{i2} + \beta_3 \cdot X_{i1} \cdot X_{i2} + \varepsilon_i$$

- We thus include an *interaction term* and approximate the CEF by a linear function from $\mathbb{R}^2 \to \mathbb{R}$
- Note: Never forget to include both variables as well as their interaction
- If we estimate a regression of this form the effect of X_{i1} on Y_i is approximately

$$\frac{\partial E\left[Y_i \middle| X_{i1}, X_{i2}\right]}{\partial X_{i1}} \approx \beta_1 + \beta_3 \cdot X_{i2}$$

- β_3 thus estimates the extent to which the effect of X_{i1} depends on X_{i2}

Python

Selecting Subsets of the Data

- Sometimes we want to use only a subset of the DataFrame, for instance if we want to run a regression only on a subset of the data
- Pandas has different methods for subset selection
- For instance, one could use the *indexing operator []* to select columns
  - `df['age']` gives back a series that contains only column age
  - `df[['age','wage']]` gives a DataFrame including only columns age & wage from the initial DataFrame df
- If we put a condition in the brackets, then rows are selected that satisfy this condition
  - `df[df['age']>50]` gives back a DataFrame containing only rows (observations) where age is larger than 50
  - We can use & (for and) and | (for or):
  - `df[(df['age']>50) | (df['age']<30)]` gives back a DataFrame that contains only observations where age<30 or >50

Python

Categorial Variables and Interaction Terms in Regressions

- For categorial variables statsmodels formulae can automatically generate dummy variables for each category with the `C()` operator:

```
smf.ols('Wage ~ age + C(Region)', data=df).fit()
```

- Interaction terms can also be directly generated with `*`

```
smf.ols('Wage ~ age * female', data=df).fit()
```

- Note: when using * statsmodels also includes the two interacted variables separately

- Furthermore: You can use functions (from numpy) to transform variables directly in the regression equation

```
smf.ols('np.log(Wage) ~ age * female', data=df).fit()
```

  – Note: the function `np.log(x)` computes the log of `x`

Your Task — Association between Management Practices & Performance

- Open your ManagementPractices.py file
- Research question: Is a management practice scoring that has been developed in one countries is equally predictive for performance in a country with a different culture?
- Background: the B/vR scoring has been developed in the UK
- Your task: Find out whether the management score is equally predictive for ROCE in China as compared to the UK
- First create a dummy variable `ChinaD` that includes only observations from China (inspect variable `country`)
- Then create a data frame that only includes data from the UK and China: `dfn=df[(df["country"]=='China')|(df["country"]=='Great Britain')]`
- Now rerun your regression of `ROCE` on management interacting `management` with `ChinaD` (do not forget to run it on the dfn DataFrame!)
- Interpret your results

2.5 Estimating Non-linear functions

- In some applications we have reason to believe that the CEF is non-linear
- For instance, wages may first increase in age and then decrease
- Many applied researchers then start by estimating a quadratic function

$$Y_i = \alpha + \beta_1 \cdot X_i + \beta_2 \cdot X_i^2 + \varepsilon_i$$

- Hence, we approximate the CEF with a quadratic function
- This can also be useful when we suspect that the CEF is concave or convex
- But be careful when interpreting β_1: this is no longer the slope parameter but

$$\frac{\partial E\left[Y_i \middle| X_i\right]}{\partial X_i} \approx \beta_1 + \beta_2 \cdot 2X_i$$

- Sign of β_2 estimates the second derivative of the function, as

$$\frac{\partial^2 E\left[Y_i \middle| X_i\right]}{\partial X_i^2} \approx 2\beta_2$$

Study **Age and Job Engagement**

- Please open again the Jupyter notebook `LPPanalysis.ipynb`
- Generate a new variable `alter2` which is `alter`^{2}
  To do so you can either compute `alter*alter` or `alter**2`
- Now regress engagement on `alter` and `alter2`
- How do you interpret the results?
- Hint: You can also graphically inspect the connection (but think about the interpretation first!) using

```
sns.regplot(y='enga_std', x='age', data=df,
            x_bins=10, order=2)
```

  – `x_bins` specifies that not each observation is plotted as a dot but neighboring observations are averaged in bins (here 10)
  – `order=2` specifies that the regression plot fits a polynomial of order 2 which is a parabola

- Sometimes researchers replace the dependent variable with its logarithm

$$ln\, Y_i = \alpha + \beta \cdot X_i + \varepsilon_i$$

- Part of reason: Logs less sensitive to outliers & may reduce heteroscedasticity (→ statistical tests)
- But more importantly: logs sometimes lead to convenient interpretations
- When X_i is a dummy variable our CEF is fully captured by a regression &
  - $ln\, Y_{i1} = \alpha + \beta + \epsilon_i$
  - $ln\, Y_{i0} = \alpha + \epsilon_i$
- Then $$\beta = ln\, Y_{i1} - ln\, Y_{i0} = ln \frac{Y_{i1}}{Y_{i0}}$$
- Such that $$\frac{Y_{i1}}{Y_{i0}} = \exp(\beta) \approx 1 + \beta$$

→ The coefficent β is approximately equal to the percentage change in the outcome variable (approximation is ok for small enough β (like $\beta < 0.2$))

→ The outcome is unaffected by the units in which Y_i is measured

Hence:

- Regression provides the best linear predictor for the dependent variable; the CEF provides the best unrestricted predictor
- Even if the CEF is non-linear, regressions provide the best linear approximation
- A/P: This "*lines up with our view of empirical work as an effort to describe essential features of statistical relationships without necessarily trying to pin them down exactly*"
- Furthermore
  - Imposing linearity reduces complexity
  - A linear function is summarized in a few parameters that often have accessible interpretations
- But: there is danger of oversimplification
  - Other machine learning techniques allow to relax assumption of linearity or on specific functional forms
  - May allow to come closer to the true CEF in complex data