People Analytics & Econometrics

The Evaluation of Management Practices

Sander Kraaij, Dirk Sliwka

Fall Term 2024

Contents

Introduction

Key questions addressed in this course:

- How can we evaluate the effect of management practices on outcome variables such as profits or job satisfaction?
- Why and when are regressions useful?
- When and how can we identify causal effects?
- How can we assess the reliability of measurement?
- How do we analyze cross-sectional and longitudinal data sets?
- How can a field experiment be set up?
- How can we set up machine learning algorithms to make predictions?

Useful Literature/Online Sources:

Econometrics & Causal Inference:

- Angrist and Pischke's Mostly Harmless Econometrics (Ch. 2 and 3) and Mastering 'Metrics: The Path from Cause to Effect
- Scott Cunningham's Causal Inference - The Mixtape
- Andrea Ichino's Lecture slides

Data Science and Econometrics with Python:

- Arthur Turell's: Coding for Economists and Python for Data Science
- Matheus Facure Alves's: Causal Inference for The Brave and True

Machine Learning with Python:

- James, Witten, Hastie, Tibshirani: An Introduction to Statistical Learning with Applications in Python
- Guido Müller's Introduction to machine learning with Python

Key distinction for study designs:

Study based on *observational* data	Laboratory experiment	Field experiment
• Data creation process not affected by the researcher • Example data: Data from surveys, balance sheets, personnel records, … • Typically no *exogenous variation* in management practices (i.e. differences in use of practices may be related to unobserved variables)	• Data generated by the researcher in the lab • Typically students are hired to make certain decisions/work • Exogenous treatment variation allows to study causal effects	• Also: RCT (Randomized Controlled Trial), or in practice A/B test • Data generated in the field (for instance in a firm) • Exogenous treatment variation allows to study causal effects

Types of Data

- To evaluate management practices, it is useful to combine different types of data
- Key sources within firms: administrative and survey data (operational vs. experience, or o-data and x-data)

Administrative data, "O-data"	Survey data, "X-data"
• Data from IT systems/personnel records on operational processes • Examples: *Quit rates, bonuses, salaries, sales, profits, hiring durations, performance evaluations, …*	• Typically generated through (online) employee surveys • Perceptions and Attitudes • Examples: *Job satisfaction, Customer satisfaction, Job engagement, commitment, …* • Also: text data from open survey questions or verbal feedback

Types of Data

Characteristics of operational/administrative data:

- Can be directly drawn from company ERP system or data warehouses
- Typically rather accurate (for instance payroll information, hiring data, …)
- But also depends on quality of processes to store subjectively assessed information (example: reasons for employee terminations)

Characteristics of survey/experience data:

- Cheap to collect through online surveys
- Measures of subjective perceptions that can be biased
- Anonymity of respondents has to be safeguarded which can make it hard to map to O-data
- Can also use population/workplace surveys (GSOEP, NLSY, LPP, MOPS, …)

0. Python Tutorial

- Now that we have been introduced to types of data, let us learn how to work with data using

1. 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

1.1 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 $X_i = x$

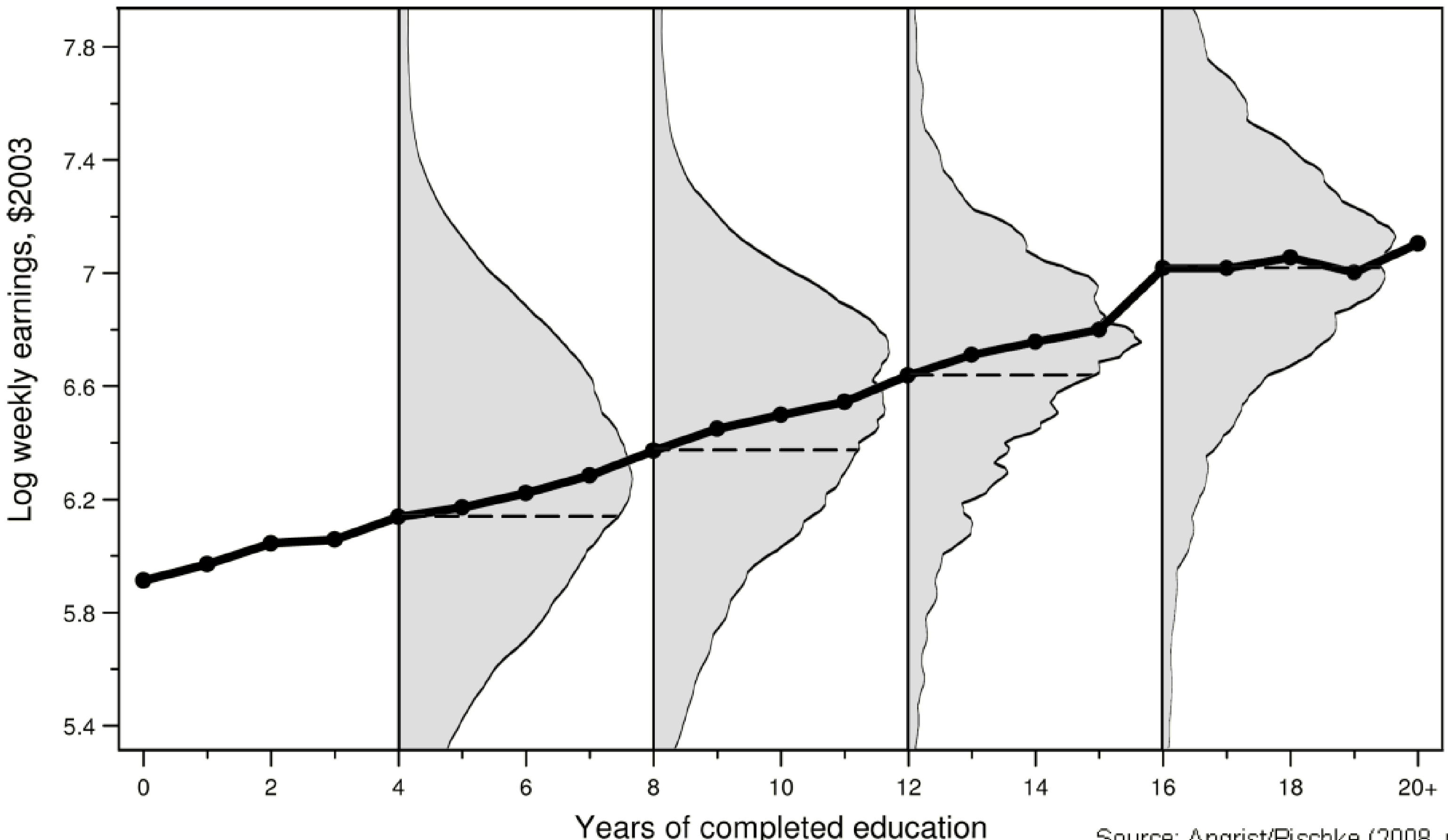
Distinguish:

- *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 $i = 1, \dots, n$
  - 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)

Python

Statistical Analyses using Python

There are several packages/modules in Python that can be used to perform statistical analyses

- *NumPy* is the underlying package for scientific computing
- *Pandas*: provides data structures
- *Statsmodels* or *PyFixest*: to perform regressions
- *Seaborn:* to visualize data with graphs

- In the beginning of our Python file we import these modules

```
import pandas as pd
import numpy as np
import pyfixest as pf
import seaborn as sns
```

- We then call functions from these modules by something like

```
df = pd.read_csv(path_to_data)
```

(Here: call function read_csv from pandas)

Python

Statistical Analyses using Python

Key concepts:

- *DataFrame* is a 2-dimensional data structure
  - Provided by Pandas
  - Like an Excel spreadsheet
  - *Columns* contain variables (example: age, wage)
  - *Rows* contain observations (example: different people)
  - The first column contains an *index* (a label for the row)
  - On the previous slide: `df = pd.read_csv(path_to_data)` reads a table from the file and stores it in a new DataFrame called df
- Missing data in a DataFrame is noted with value `NaN`
- A *Series* is like a list containing one variable (also has an *index)*

Python

Printing Summary Statistics

- We typically start an analysis by looking at descriptive statistics
  - What are the means of the key variables?
  - What are their standard deviations?
  - How are specific variables correlated?
- To print summary statistics, use the describe() method
  - df.describe() prints summary statistics for all variables
  - df['varname'].describe() or df.varname.describe() prints summary statistics for variable varname
- Or we can directly compute the mean or standard deviation with df.varname.mean() and df.varname.std()
- We can also explore summary statistics for specific subgroups (rows) df.groupby('country').varname.describe()

Python

Graphs in Python

- It is often useful to visualize data with graphs
- Particularly useful & easy to use: 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 categorical variables
  - Adds confidence bands: from all samples that can be drawn, the confidence interval will contain the true population mean in 95% of the cases (more about this in chapter 3)
- sns.relplot(x='income', y='happiness', data=df)
  - Scatter plot where each dot is a data point
- sns.histplot(df['wage'])
  - Plots histogram of the variable
  - Note: df['x'] returns a series of all observations of variable x

Your Task

Feedback Talks and Job Satisfaction

- Analyze data from the LPP, a matched employer-employee survey data set for Germany (see [Kampkötter et al. (2016)]) which combines
  - An establishment survey on HR practices
  - An employee survey on HR practices and attitudes
- We can access a campus file generated by IAB for teaching purposes that matches the two data sets for a subset of firms and employees
- Variables from the establishment survey start with a *b*, those from the employee survey with an *m*
- Files:
  - https://raw.githubusercontent.com/dsliwka/EEMP2024/refs/heads/main/Data/LPP-CF_1215_v1.csv (CSV format version of the data set)
  - https://github.com/dsliwka/EEMP2024/blob/main/Data/VariablesLabelsLPP.pdf (short English variable description)
  - http://doku.iab.de/fdz/reporte/2017/DR_09-17.pdf (detailed documentation; unfortunately only in German)

Your Task

Feedback Talks and Job Satisfaction

- Create a new Colab notebook and import packages
  - import pandas as pd
  - import numpy as np
  - import seaborn as sns
- Read the data (subset of the data for teaching purposes) into a DataFrame
  - path_to_data = 'https://raw.githubusercontent.com/dsliwka/EEMP2024/refs/heads/main/Data/LPP-CF_1215_v1.csv'
  - df = pd.read_csv(path_to_data)
- Inspect the data with describe
- Look at the employees' job satisfaction (for instance plot a histogram):
  - msat_job gives you the job satisfaction stated in the survey
- What is the share of employees who have an annual feedback interview?
  - mmagespr is a dummy which has value 1 if the employee had a feedback interview with his/her boss last year.
- Save your notebook as LPPanalysis.ipynb

Your Task

Feedback Talks and Job Satisfaction

- Let us use the LPP to study the association between the use of feedback interviews and employee engagement

- To estimate the CEF, simply compare the mean of job satisfaction between employees who had a feedback interview and those who didn't
  - msat_job gives you the job satisfaction stated in the survey
  - mmagespr is a dummy variable which is equal to 1 if the employee had a feedback interview and 0 otherwise
  - 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. 32-33)

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 with ("orthogonal to") 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

1.2 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

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

- 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

We can do the same in the **multivariate case**

- We can approximate the CEF with a multivariate linear function

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_k X_{ik}$$

- Proceeding analogously to the bivariate case, we obtain (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* $\hat{\beta} = \frac{\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y})}{\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})^2}$

Note:

- Proceed in the same way in the multivariate case (not shown here)

Python

OLS Regressions in Python: The PyFixest Package

- Several Python packages which you can use to estimate regression: (most prominent are Statsmodels and Linearmodels)
- We will use a new and very convenient package PyFixest
  - Install the latest version:

    ```
    !pip install pyfixest -q
    ```
- This is a Python implementation of package Fixest in R. Hence, when switching to R, you will be able to use essentially the same syntax.
- Basic use: If you have a DataFrame df containing variables y, x1 and x2
  - To regress y (dependent variable) on x1 and x2 (indep. variables):

    ```
    reg = pf.feols ('y ~ x1 + x2', data=df)
    ```
  - And show the results with one of:

    ```
    reg.summary()
    reg.tidy()
    ```

Multiple Regressions with PyFixest

- It is often much more convenient to display tables with different specifications side by side with etable:

  ```
  reg1 = pf.feols('y ~ x1', data=df)
  reg2 = pf.feols('y ~ x1 + x2', data=df)
  pf.etable ([reg1, reg2])
  ```

- feols **has built-in *stepwise* functions with which you can quickly estimate multiple regressions in one line of code:** sw, sw0, csw, csw0 (“0” **is a zero)**
  - regs = pf.feols('y ~ x1 + sw(x2, x3)', data=df) **will directly run two regressions** y ~ x1 + x2 **and** y ~ x1 + x3 **and return their results**
  - regs = pf.feols('y ~ x1 + sw0(x2, x3)', data=df) **will also include** y ~ x1
  - regs = pf.feols('y ~ x1 + csw(x2, x3)', data=df) **will do this cumulatively, i.e. regress** y ~ x1 + x2 **and** y ~ x1 + x2 + x3
- **You can display all regressions results with** pf.etable (regs)

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

Study

Observational Data: Management Practices and Performance

Bloom and 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 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/dsliwka/EEMP2024/refs/heads/main/Data/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.histplot(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

- Compare the management scores between different countries
  - Convenient to generate Dataframe with scores aggregated by country:

    dfagg = df.groupby('country').management.mean().reset_index()

    Note: reset_index() adds country as variable (otherwise it is the index)

  - Then you can sort this Dataframe with dfagg = dfa.sort_values()

  - And plot it with sns.barplot

    Note: Better use country as y variable and management as x

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

- Now run a regression of roce as dependent variable on management
  - Recall the syntax (adapt!):
  - reg = pf.feols ('yvar ~ xvar1 + xvar2', data=df)
    etable(reg)
- In a second step, add the number of employees emp and the firm's capital ppent as control variables in a second regression and show them side-by-side
- Interpret your result
- Save your notebook as ManagementPractices to reuse it later

Python

Table Layout with PyFixest etable

- You can improve the layout of the regression table
- For instance, when you want to give your variables different names, you can create a dictionary with labels
- ```
  labels = {"management": "Management Score",
          "roce": "ROCE",
          "emp": "Number of Employees",
          "ppent": "Capital"}
  ```
- And then pass the dictionary to etable

  ```
  pf.etable([reg1,reg2], labels=labels)
  ```
- For more examples see https://py-econometrics.github.io/pyfixest/table-layout.html
  ```

1.3 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, 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

- Open again the notebook LPPanalysis.ipynb
- Estimate a regression of job satisfaction on the mmagespr dummy
- Compare the constant term (intercept) and coefficient of mmagespr with the means for the two groups computed in the last exercise.
  What do you see?
- Inspect the robustness of the connection between job satisfaction 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)
  - Working from home (dummy mheim)
  - Training (dummy mwb)

Python **Generate Dummy Variables**

- Dummy variables are variables that take only values 1 or 0
- A simple way to generate a dummy variable is to use a Boolean expression
  df['old'] = (df.age>50)
  - This generates a Boolean variable which takes the value True when the condition holds and otherwise False
  - If you want a numerical dummy, just write df['old'] = (df.age>50)*1
    (It is common to use the numerical 0/1 coding)
- If you have a categorical variable (such as country) that contains multiple values you can also use the get_dummies method in Pandas:
  - df = pd.get_dummies(df, columns=['country'])
  - This adds a dummy variable for each country and names the variables country_Australia, country_Brazil, …

1.4 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 $Y_i = \alpha + \beta_1 X_{i1} + \beta_2 X_{i2}$
- Then we may want to allow for the possibility that the effect of X_{i1} depends on the value 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 *heterogeneous 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 $\mathbb{R}^2 \rightarrow \mathbb{R}$

- 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

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

- 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]` returns 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)]` returns a DataFrame that contains only observations where age<30 or >50

Python

Categorical Variables and Interaction Terms in Regressions

- For categorical variables, PyFixest can automatically generate dummy variables for each category with the C() operator:

  ```
  pf.feols('Wage ~ age + C(Region)', data=df)
  ```

- Interaction terms can also be directly generated with *

  ```
  pf.feols('Wage ~ age * female', data=df)
  ```

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

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

  ```
  pf.feols('np.log(Wage) ~ age * female', data=df)
  ```

  Note: the function np.log(x) computes the natural logarithm of x

Python

Illustrate Patterns in Data

- When inspecting categorical variables, add a third dimension to a barplot:
  sns.barplot(x='country’, y='income', hue='gender', data=df)
- When inspecting the connection between continuous and categorical variables, you can plot different regressions on top of each other, for instance to see how a relationship looks in subsamples defined by the categorical variable:

```
sns.regplot(x='xvar', y='yvar', data=df[df.year==2005])
sns.regplot(x='xvar', y='yvar', data=df[df.year==2008])
```

- regplot has further convenient options:
  – Instead of plotting each data point, you can create bins:
    x_bins=10 for instance specifies that not each observation is plotted as a dot but neighboring observations are averaged in bins (here 10)
  – You can turn off the scatter plot with scatter=False

Your Task — Association between Management Practices & Performance

- Open your ManagementPractices notebook
- Research question: Is a management practice scoring that has been developed in one country is equally predictive for performance in a country with a different culture?
- Background: the B/vR scoring has been developed in Great Britain (GB)
- Your task: Find out whether the management score is equally predictive for ROCE in China as compared to GB
- First create a dummy variable China that indicates whether an observation is from China (inspect variable country)
- Then create a data frame that only includes data from GB and China:
  - Here it is convenient to use the x.isin(list) method that checks whether a variable (here x) is in a list (list) such as

```
dfn = df[(df.colour.isin(['blue', 'green'])]
```

Your Task — Association between Management Practices & Performance

- Now work with the smaller `dfn` DataFrame you just created
- First compare the management score between China and Great Britain
- And regress `roce` on `management` using the smaller Dataframe
- Now add an interaction term between `management` and the `China` dummy
- Note: It is instructive to compare these two regressions with two further regressions (put all four in one table with `etable`)
  - only with Chinese data
  - only with the British data

  Recall you can run a regression on a subset of the data with

  ```
  feols('y ~ x', data=df[df.colour=='blue'])
  ```
- Interpret your results

1.5 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 sign of the second derivative of the function, as

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

Your Task — Age and Job Satisfaction

- Open again the 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 job satisfaction 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='msat_job', 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 estimates 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 are less sensitive to outliers and may reduce heteroscedasticity (→ statistical tests)
- But also: logs sometimes lead to convenient interpretations
- When X_i is a dummy variable, our CEF is fully captured by a regression:
  - $\ln Y_{i1} = \alpha + \beta + \varepsilon_i$
  - $\ln Y_{i0} = \alpha + \varepsilon_i$
- 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 coefficient β is approximately equal to the percentage change in the outcome variable (approximation is okay for small enough β (like $\beta < 0.2$))

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

Mean Squared Error and the Coefficient of Determination R^2

- We can use our regression to make predictions
  (much more on this in chapter 7 on Machine Learning)
- To so we use our estimates to predict Y based on X
  - We can do so by computing the prediction $\hat{Y}_i = \alpha + \hat{\beta}_1 X_{i1} + \hat{\beta}_2 X_{i2} \dots$
  - This gives us an estimate of Y for specific values of $X_1, X_2, \dots$
- Sometimes we are thus interested in the predictive power of our regression
- Useful starting point is often the **mean squared error (MSE)**
- That is, the average squared deviation between actual values of Y and predicted values $\hat{Y}$

$$MSE = \frac{1}{N} \sum_{i=1}^{N} \left(Y_i - \hat{Y}_i\right)^2$$

Mean Squared Error and the Coefficient of Determination R²

The coefficient of determination R² is the proportion of the variance in the dependent variable that is predictable from the independent variables

$$R^2 = 1 - \frac{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2}{\frac{1}{N}\sum_{i=1}^{N}(Y_i - \bar{Y})^2} = 1 - \frac{MSE}{V[Y]}$$

where $\bar{Y} = \frac{1}{N}\sum_{i=1}^{N} Y_i$ is the mean of the Y_i

- When the prediction is perfectly accurate the $MSE = 0$ and $R^2 = 1$
- When it is completely inaccurate the best prediction is $\hat{Y}_i = \bar{Y}$ and then $R^2 = 0$
- Note: When the R^2 is small, the regression…
  - … is likely not useful to make good predictions of Y
  - … but can still (sometimes) be useful to estimate the impact of a specific variable – provided that we estimate this impact precisely
  - This is what we look at in the next chapters

Summary:

- 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 the assumption of linearity or specific functional forms
  - May allow to come closer to the true CEF in complex data