People Analytics & Econometrics

The Evaluation of Management Practices

Sander Kraaij, Dirk Sliwka

Fall Term 2023

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?
- How can we assess the reliability of measurement?
- Why and when are regressions useful?
- When and how can we identify causal effects?
- 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:

- Angrist and Pischke (2009): Mostly Harmless Econometrics: An Empiricist's Companion, Chapters 2 and 3
- Angrist and Pischke (2014): Mastering 'Metrics: The Path from Cause to Effect
- Wooldridge (2003) (Background reading)
- James, Witten, Hastie, Tibshirani (2017): An Introduction to Statistical Learning with Applications in R
- Müller, Guido (2016) Introduction to machine learning with Python: a guide for data scientists
- Andrea Ichino’s lecture slides (for some links to standard econometrics courses): http://www.andreaichino.it/teaching-material

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. Survey Data and Scale Reliability

- In surveys, we can ask people how they feel, or about their own perceptions about behavior
- This is mostly done through *survey items* that the respondent is asked to evaluate, such as “I am very satisfied with my job”
- In commonly used *Likert-scales*, respondents are asked for their level of agreement on a number of given statements on a scale such as
    1. Strongly disagree
    2. Disagree
    3. Neither agree nor disagree
    4. Agree
    5. Strongly agree
- While practitioners are often tempted to use single items for a certain attitude or behavior, researchers stress the importance to use *multiple items* to assess a phenomenon

Psychological Constructs and Reliability

- Researchers typically use scales with *multiple items* that are supposed to measure certain psychological constructs
- A *psychological construct* is a label for a cluster of covarying behaviors or attitudes (such as job satisfaction, job engagement, but also of personality traits such as conscientiousness, extraversion, etc.)
- Typically
  - item responses are added up to a score
  - the score then represents a person's position on the construct
- Important question: how *reliable* is a scale?
- That is, how consistently does a scale measure the same underlying construct?

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*: 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 statsmodels.api as sm
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

Looking at Correlations

- In a next step, researchers often inspect the correlation between variables
- We can obtain a correlation matrix with `df.corr()`
  - Note: this can be a huge matrix as it shows the correlation coefficients between all variables in the DataFrame
  - Typically, it makes sense to only show it for a subset of the data
- To do so, we can filter the data frame (which gives us a smaller data frame selected by the filtering criteria)
  - Show correlation between two variables age and tenure:
    ```
    df.filter(items=['age', 'tenure']).corr()
    ```
  - Show correlation matrix for all variables starting with “Satis”:
    ```
    df.filter(regex='Satis*').corr()
    ```

Your Task

Analyze Survey Data

- 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/EEMP2023/main/Data/LPP-CF_1215_v1.csv (CSV format version of the data set)
  - https://github.com/dsliwka/EEMP2023/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 **Analyze Survey Data**

- Create a new Colab notebook and import packages
  - `import pandas as pd`
  - `import numpy as np`
- Read the data (subset of the data for teaching purposes) into a DataFrame
  - `path_to_data = "https://raw.githubusercontent.com/dsliwka/EEMP2023/main/Data/LPP-CF_1215_v1.csv"`
  - `df = pd.read_csv(path_to_data)`
- Inspect the data with `describe`
- What is the share of employees who have an annual appraisal interview?
  - Use the variable `mmagespr`. This is a dummy which has value 1 if the employee had an appraisal/feedback interview with his/her boss last year.

Your Task

Analyze Survey Data

- The data set includes a scale to measure employee engagement, a short version of the Utrecht Work Engagement Scale (Schaufeli et al. (2006)):
  - *At my work, I feel bursting with energy*
  - *At my job, I feel strong and vigorous*
  - *I am enthusiastic about my job*
  - *My job inspires me*
  - *When I get up in the morning, I feel like going to work*
  - *I feel happy when I am working intensely*
  - *I am proud of the work that I do*
  - *I am immersed in my work*
  - *I get carried away when I'm working*
- The response scale ranges from 1 "every day" to 5 "never"
- The respective 9 item variables in the data set start with `menga`
- Print the correlation matrix for these variables
- Save the notebook for later use (name it `LPPanalysis.ipynb`)

Assessing Reliability: Classical Test Theory in Psychology

- Assumption: Response = sum of the "true score" T of the construct & noise

$$X = T + \varepsilon$$

- But how noisy is our measure?
- Consider the share of the variance of X due to the variance of T

$$\frac{\sigma_T^2}{\sigma_X^2} = \frac{\sigma_T^2}{\sigma_T^2 + \sigma_\varepsilon^2}$$

- The higher this *reliability coefficient*, the less noisy is the measurement
- Note that the correlation coefficient between X and T is equal to

$$\rho_{XT} = \frac{Cov[X,T]}{\sigma_X \sigma_T} = \frac{Cov[T+\varepsilon,T]}{\sigma_X \sigma_T} = \frac{\sigma_T^2}{\sigma_X \sigma_T} = \frac{\sigma_T}{\sigma_X}$$

- Hence, the reliability coefficient is often denoted as ρ_{XT}^2, i.e. the squared correlation between true and observed score
- But note: with a single item we cannot measure ρ_{XT}^2 as we do not know T

The Reliability of Scales

- But suppose we have more items that all measure the same construct
- This has two key advantages
  - We can *assess how reliable* the scale is
  - Using average response across items makes measurement *more reliable*

To see the former: Suppose two items, X_1 and X_2, measure the construct

- Assume that both assess the same true score, and their noise is independently drawn from the same distribution (note: these are strong assumptions!)

- We can now estimate from the responses to our survey

$$\rho_{X_1 X_2} = \frac{Cov[X_1, X_2]}{\sigma_{X_1}\sigma_{X_2}} = \frac{Cov[T + \varepsilon_1, T + \varepsilon_2]}{\sigma_X^2} = \frac{\sigma_T^2}{\sigma_X^2}$$

$\rightarrow$ The correlation between the two item responses then gives a measure of the reliability of *each of the two items separately* (not of the two-item scale)

The Reliability of Scales: Length of the Scale

- Suppose now we have $i = 1, \dots, k$ items that measure the construct
- The response to each item i is $X_i = T + \varepsilon_i$
- Consider the average score $\bar{X} = \frac{1}{k}\sum_{i=1}^{k} X_i$
- Now compute the reliability of $\bar{X}$

$$\rho^2_{\bar{X}T} = \frac{V[T]}{V[\bar{X}]} = \frac{V[T]}{V[\frac{1}{k}\sum_{i=1}^{k}(T + \varepsilon_i)]}$$

$$= \frac{\sigma_T^2}{V[T + \frac{1}{k}\sum_{i=1}^{k}\varepsilon_i]} = \frac{\sigma_T^2}{V[T] + V[\frac{1}{k}\sum_{i=1}^{k}\varepsilon_i]}$$

$$= \frac{\sigma_T^2}{\sigma_T^2 + \frac{1}{k^2}k\sigma_\varepsilon^2} = \frac{\sigma_T^2}{\sigma_T^2 + \frac{1}{k}\sigma_\varepsilon^2}$$

→ The reliability thus increases in the length of the scale k

The Reliability of Scales: Cronbach's Alpha

- Consider again the reliability coefficient $\rho^2_{\bar{X}T} = \frac{V[T]}{V[\bar{X}]}$
- Note that for any two items, we have that

$$Cov[X_1, X_2] = Cov[T + \varepsilon_1, T + \varepsilon_2] = V[T]$$

- Now estimate $V[T]$ by the mean of all covariances between any two items:

$$\overline{\sigma_{ij}} = \frac{1}{k(k-1)} \sum_{i=1}^{k} \sum_{j \neq i}^{k} Cov[X_i, X_j]$$

- The ratio $\rho^2_{\bar{X}T} = \frac{\overline{\sigma_{ij}}}{V[\bar{X}]}$ is called *Cronbach's alpha*
- It can be rearranged to become

$$\rho^2_{\bar{X}T} = \frac{\overline{\sigma_{ij}}}{V[\bar{X}]} = \frac{k}{k-1}\left(1 - \frac{\sum_{i=1}^{k} V[X_i]}{V\left[\sum_{i=1}^{k} X_i\right]}\right)$$

The Reliability of Scales: Cronbach's Alpha

Cronbach's alpha

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\sum_{i=1}^{k} V[X_i]}{V\left[\sum_{i=1}^{k} X_i\right]}\right)$$

Note:

- α is a very frequently applied measure for the *internal consistency* of a scale
- A scale is for instance considered to have a good internal consistency if $\alpha > 0.8$
- Cronbach's alpha gives a lower boundary to reliability (as the derivation used the assumption that all items assess the same true score and their noise is independently drawn from the same distribution)

Python

Computing Item Averages and Standardizing

- To obtain the score for the scale we typically compute the average across all items of the scale
- In Python we can do that for instance (say we have four items measuring satisfaction called `satis1, …, satis4`)
  - by "manually" summing up the items and averaging:
    `df['satis'] = (df.satis1+df.satis2+…) / 4`
  - or averaging across all columns of a filtered DataFrame:
    `df['satis'] = df.filter(regex='satis*').mean(axis=1)`
  - Note: the method `mean` returns the mean of the values either over rows/observations (`axis=0`) or columns/variables (`axis=1`)
- Frequently, scores are *standardized* $X_{STD} = \frac{X - m_X}{\sigma_X}$ where m_X is the mean and σ_X the standard deviation of X
- We can do that for instance by
  `df['sat_std'] = (df.satis-df.satis.mean())/df.satis.std()`

Python

Computing Cronbach's alpha

- We can use method `cronbach_alpha` from package `pingouin`
  - To so we must first install `pingouin` with `!pip install pingouin`
  - Then we can `import pingouin as pg`
  - You call the function with `pg.cronbach_alpha(data=df)`
- Or we can define our own function:

```
def cronbach(data):
  k = data.shape[1]
  varX = data.sum(axis=1).var()
  sumVar = data.var(axis=0).sum()
  return k / (k-1) * (1 - sumVar/varX)
```

- Note: the DataFrame you pass to either function must only consist of the variables of the specific scale, you can generate such a DataFrame with `df.filter(regex='menga*')`

Your Task — **Estimate the Reliability of a Scale**

- Please open again the notebook `LPPanalysis.ipynb` used to look at the engagement data
- Generate a new variable `enga` for the mean engagement score
- Note: As the variable is coded, low values indicate high engagement. To avoid later confusion, it makes sense to reverse the scale (you can do that by simply stating `enga = 6-enga`)
- Also generate a standardized version of the variable (call it `enga_std`)
- What is the value of Cronbach's alpha? To what extent is the engagement scale internally consistent?