POLITECNICO MILANO 1863

Exercise Session - Multivariate statistics

AY 2024/2025 - Gloria Peggiani

PCA & Cluster Analysis

01

Exercise 1: Young People Survey

Exercise: Young People Survey

Objectives Definition

Data Gathering & Integration

Data Exploration & Data Preparation

Cluster Analysis

Objective: to investigate music preferences of young people

Data collection: a large-scale survey among young people was conducted

Data exploration: descriptive analysis was performed

Data preparation: anomalies and inconsistencies have been identified and removed. PCA allows to reduce the dimensionality of the dataset.

Preliminary Operations

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import seaborn as sns
import matplotlib.pyplot as plt
from factor_analyzer import Rotator

file_path = r"C:\Users\gp\Desktop\06_Exercise 1 dataset.xlsx"
df = pd.read_excel(file_path)

df.head()

print(df.describe())
```

	ID	Music	Fast	Dance	Folk	Country	Classical music	Musical	Pop	Rock	...	Alternative	Latino	Techno	Opera
0	1	5	3	2	1	2	2	1	5	5	...	1	1	1	1
1	2	4	4	2	1	1	1	2	3	5	...	4	2	1	1
2	3	5	5	2	2	3	4	5	3	5	...	5	5	1	3
3	4	5	3	2	1	1	1	1	2	2	...	5	1	2	1
4	5	5	3	4	3	2	4	3	5	3	...	2	4	2	2

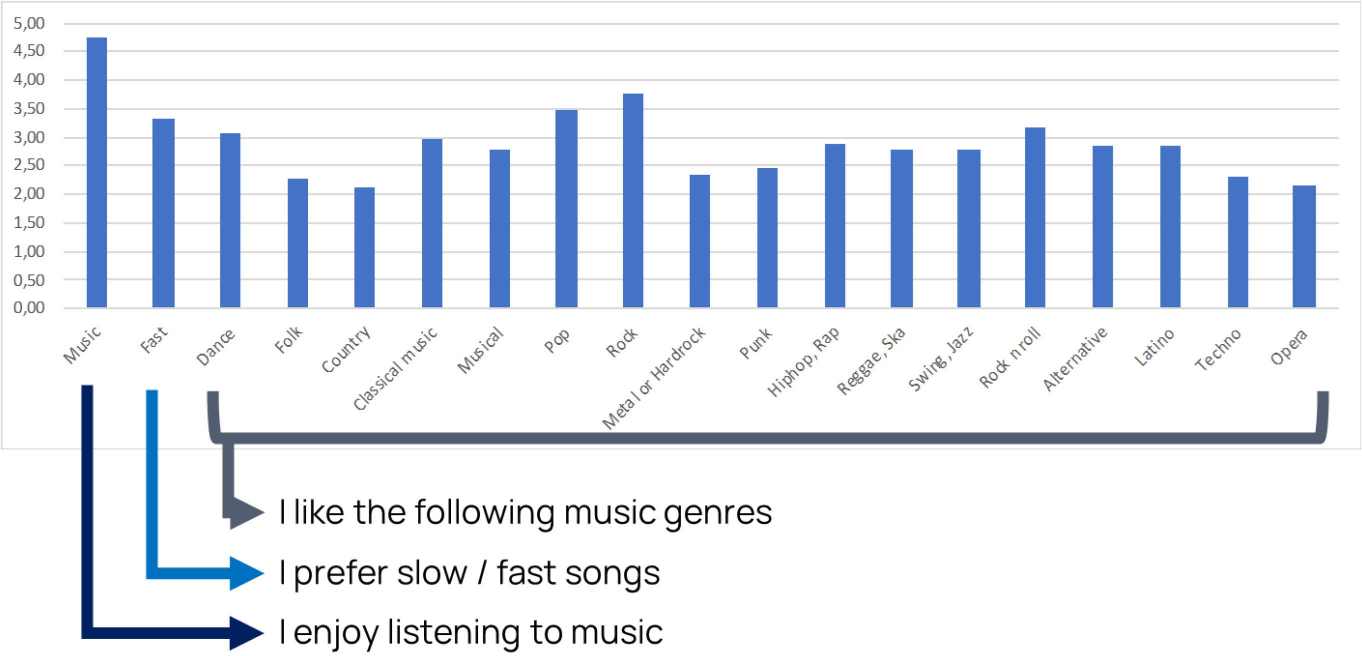
```
               ID       Music        Fast       Dance        Folk     Country
count  853.000000  853.000000  853.000000  853.000000  853.000000  853.000000
mean   485.966002    4.740914    3.322392    3.086753    2.283705    2.137163
std    284.226642    0.653397    0.818178    1.173645    1.144154    1.083894
min      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000
25%    239.000000    5.000000    3.000000    2.000000    1.000000    1.000000
50%    481.000000    5.000000    3.000000    3.000000    2.000000    2.000000
75%    736.000000    5.000000    4.000000    4.000000    3.000000    3.000000
max    982.000000    5.000000    5.000000    5.000000    5.000000    5.000000
```

PCA: Steps

- Variables selection
- Rotation method identification
- Number of principal component definition
- Results interpretation

Principal Component Analysis

- **Variables selection**

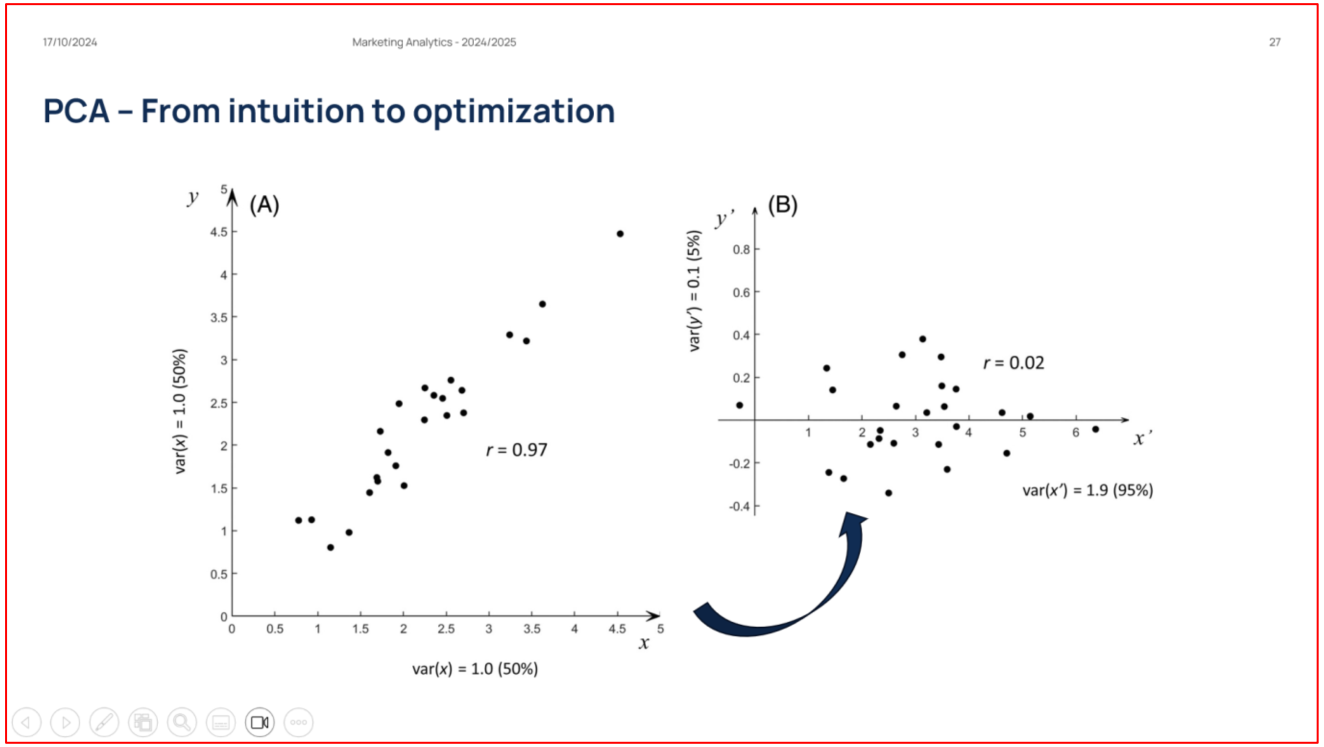



```
pca_columns = [
    'Dance', 'Folk', 'Country', 'Classical music', 'Musical', 'Pop', 'Rock',
    'Metal or Hardrock', 'Punk', 'Hiphop, Rap', 'Reggae, Ska', 'Swing, Jazz',
    'Rock n roll', 'Alternative', 'Latino', 'Techno', 'Opera'
]
df_pca = df[pca_columns]
```

Principal Component Analysis

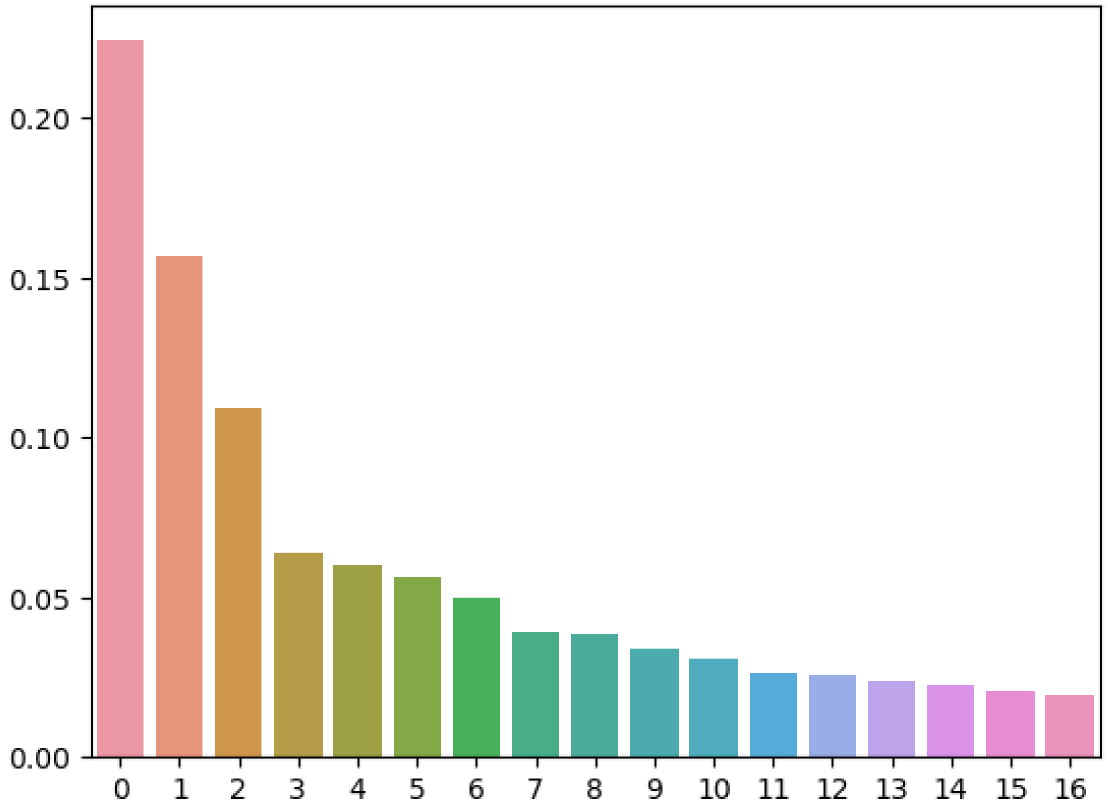
```
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_pca)

pca = PCA(n_components=17)
pca.fit(df_scaled)
```



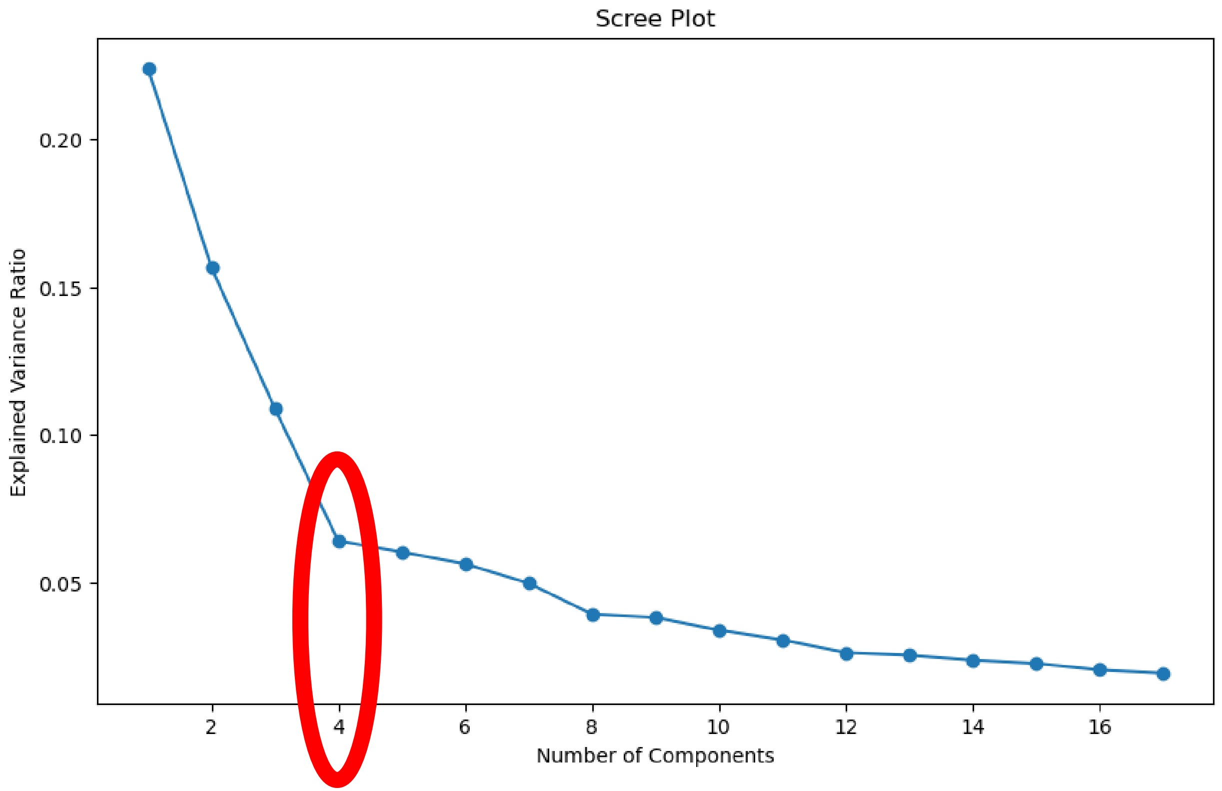
Principal Component Analysis

```
explained_var=pd.DataFrame(pca.explained_variance_rat
io_).transpose()
%matplotlib inline
import seaborn as sns
ax = sns.barplot( data=explained_var)
```



Principal Component Analysis

```
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(pca.explained_variance_ratio_)
+ 1), pca.explained_variance_ratio_, marker='o')
plt.xlabel('Number of Components')
plt.ylabel('Explained Variance Ratio')
plt.title('Scree Plot')
plt.show()
```



Principal Component Analysis

- **Number of principal component definition**

```
pca = PCA(n_components=5)
df_pca_5 = pca.fit_transform(df_scaled)

components = pca.components_
```

Principal Component Analysis

- **Rotation method identification**

```
rotator = Rotator(method='varimax')
rotated_components =
rotator.fit_transform(components.T).T
```

The goal of component rotation is to improve the interpretability of the factor solution by reaching simple structure

Orthogonal rotation (Varimax):
assumes that components are independent or uncorrelated with each other;

Oblique rotation (Oblimin):
assumes that components are not independent and are correlated

```
rotated_df = pd.DataFrame(rotated_components,
index=['PC1', 'PC2', 'PC3', 'PC4', 'PC5'],
columns=df_pca.columns)

output_file_path_rotated = r"C:\Users\gp\Desktop\rotated_components.xlsx"
rotated_df.to_excel(output_file_path_rotated, index=True)
```

Principal Component Analysis

- **Adding the selected PCs to the original dataset**

```
df_pca_5_df = pd.DataFrame(df_pca_5, columns=['PC1', 'PC2', 'PC3',
'PC4', 'PC5'])
df_with_pca = pd.concat([df, df_pca_5_df], axis=1)
```

```
output_file_path_updated = r"C:\Users\gp\Desktop\df_with_pca.xlsx"
df_with_pca.to_excel(output_file_path_updated, index=False)
```

K-Means clustering: Steps

- Variables selection
- Number of clusters identification
- Convergence assessment
- Robustness assessment
- Results interpretation

Preliminary Operations

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.formula.api import ols

file_path = r"C:\Users\gp\Desktop\06_Exercise 1 dataset_cluster.xlsx"
df = pd.read_excel(file_path)
```

```
df.head()
```

Pop	Rock	...	Height	Weight	Siblings	Gender	Residence	Classy	Rocky	Dancy	Disco	Jazzy
5	5	...	163	48	1	2	2	-1.27514	0.17886	0.75151	-1.27695	-2.03313
3	5	...	163	58	2	2	1	-1.71585	1.65905	-0.18118	-1.21665	-0.37864
3	5	...	176	67	2	2	1	0.70616	1.15932	0.75922	-1.60703	0.87778
2	2	...	172	59	1	2	1	-1.51192	-0.43809	-1.97382	-0.41117	-0.05444
5	3	...	170	59	1	2	2	0.10928	-1.11273	0.90231	0.58155	-0.02226

K-Means clustering

- **Variables selection**

```
kmeans_columns = ['Classy', 'Rocky', 'Dancy', 'Disco', 'Jazzy']
df_kmeans = df[kmeans_columns]
```

```
scaler = MinMaxScaler()
df_scaled = scaler.fit_transform(df_kmeans)
```

K-Means clustering

```
km = KMeans(n_clusters=5,
            init='random',  # init='k-means++'
            n_init=10,
            max_iter=300,
            tol=1e-04,
            random_state=0)
```

```
km.fit(df_scaled)
```

We predict the closest cluster each observation belongs to:

```
y_km = km.predict(df_scaled)
```

K-Means clustering

We add the cluster information to the initial dataset:

```
df['Cluster'] = y_km

df.head()
```

...

Residence	Classy	Rocky	Dancy	Disco	Jazzy	Cluster
2	-1.27514	0.17886	0.75151	-1.27695	-2.03313	3
1	-1.71585	1.65905	-0.18118	-1.21665	-0.37864	0
1	0.70616	1.15932	0.75922	-1.60703	0.87778	0
1	-1.51192	-0.43809	-1.97382	-0.41117	-0.05444	0
2	0.10928	-1.11273	0.90231	0.58155	-0.02226	3

K-Means clustering

- **Convergence assessment**

```
print(f"Convergence reached: {km.n_iter_} iterations")
```

- **N° of cases in each cluster**

```
cluster_counts = df['Cluster'].value_counts().sort_index()
print("Number of cases in each cluster:")
print(cluster_counts)
```

K-Means clustering

- **Robustness assessment**

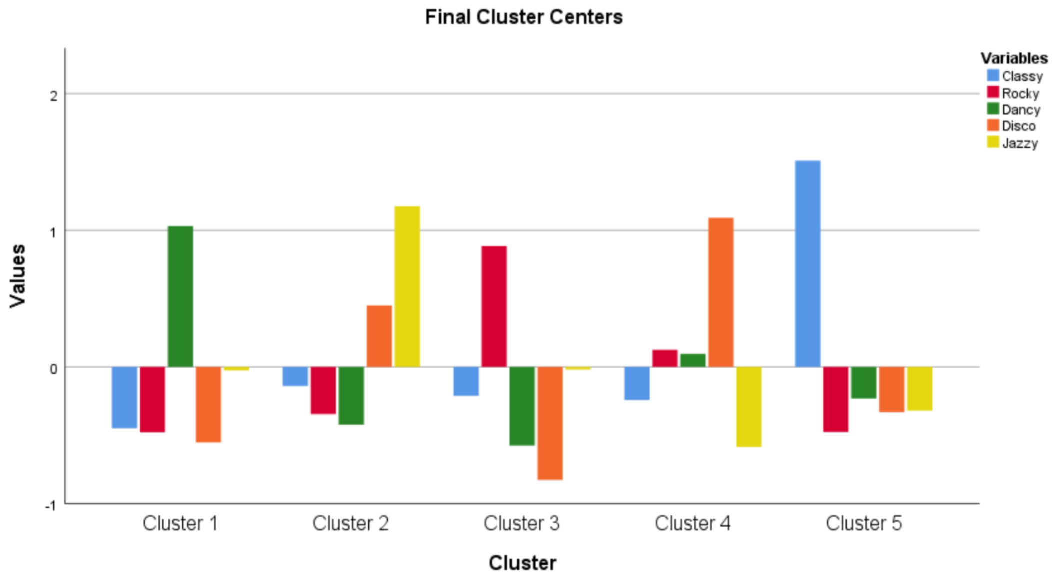
```
anova_results = {}
for column in kmeans_columns:
    model = ols(f'{column} ~ C(Cluster)',
data=df).fit()
    anova_table = sm.stats.anova_lm(model, typ=2)
    anova_results[column] = anova_table
```

```
for column, anova_table in anova_results.items():
    print(f'ANOVA table for {column}:\n',
anova_table)
```

```
ANOVA table for Classy:
                 sum_sq     df           F         PR(>F)
C(Cluster)  405.463893    4.0  192.500284  2.186987e-117
Residual    446.536200  848.0         NaN            NaN
ANOVA table for Rocky:
                 sum_sq     df           F         PR(>F)
C(Cluster)  443.588761    4.0  230.260298  8.806888e-134
Residual    408.410909  848.0         NaN            NaN
ANOVA table for Dancy:
                 sum_sq     df          F        PR(>F)
C(Cluster)  259.224505    4.0  92.708966  2.055996e-65
Residual    592.775409  848.0        NaN           NaN
ANOVA table for Disco:
                 sum_sq     df           F         PR(>F)
C(Cluster)  464.682864    4.0  254.346558  1.584314e-143
Residual    387.317084  848.0         NaN            NaN
ANOVA table for Jazzy:
                 sum_sq     df          F        PR(>F)
C(Cluster)   68.480896    4.0  18.529161  1.305245e-14
Residual    783.519019  848.0        NaN           NaN
```

K-Means clustering

- **Results interpretation**




```
cluster_means = df.groupby('Cluster')[kmeans_columns].mean()

cluster_means.plot(kind='bar', figsize=(12, 8))
plt.xlabel('Cluster')
plt.ylabel('Average Value')
plt.legend(title='Variables')
plt.grid()
plt.show()
```

PLS-SEM - Exercises

05

Exercise 3: Corporate Reputation

Exercise: Corporate Reputation

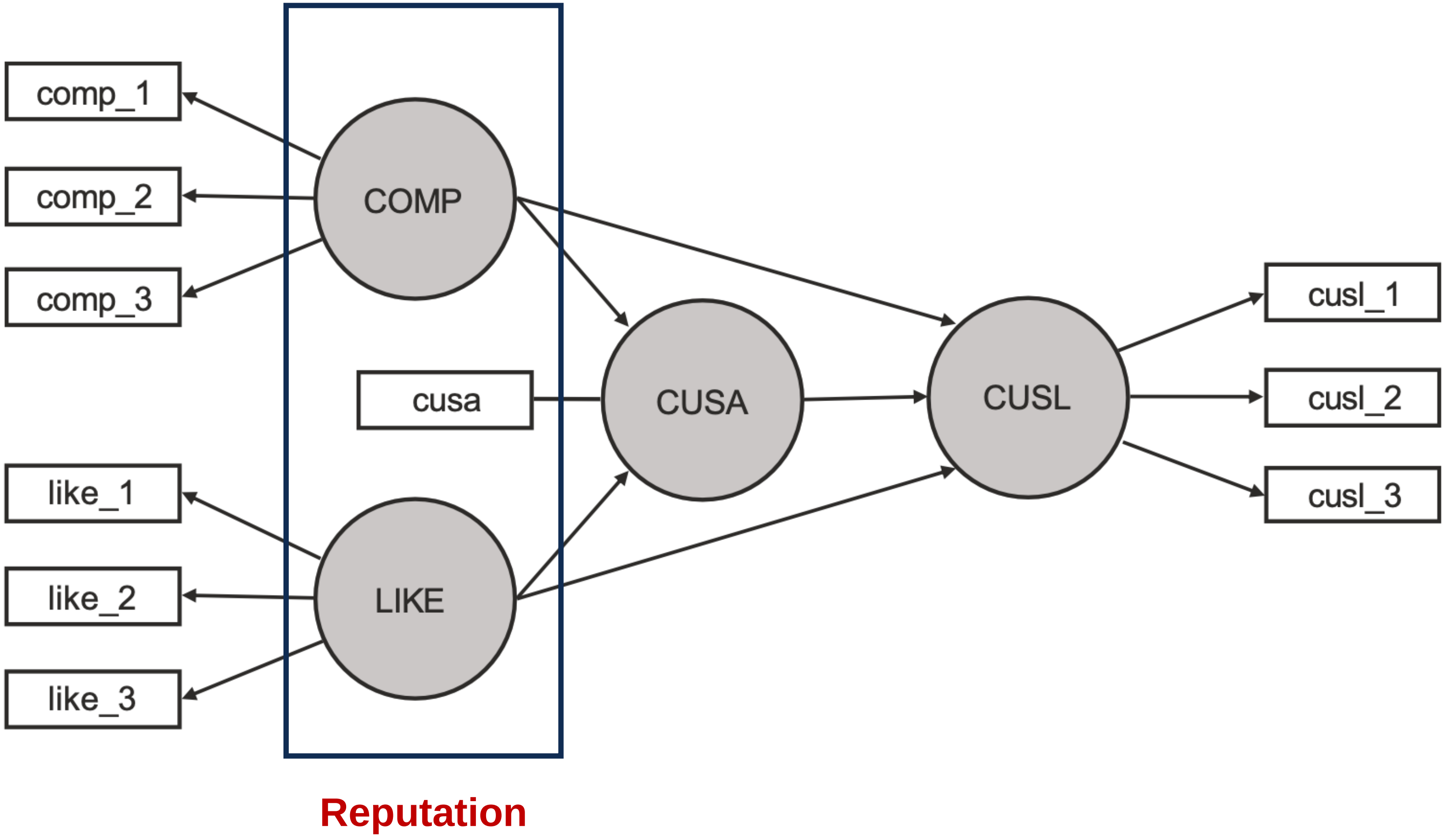
Objective: explain the effects of **corporate reputation** on **customer satisfaction** (CUSA) and, ultimately, **customer loyalty** (CUSL).

Corporate reputation represents a company's overall evaluation by its stakeholders. This construct is typically measured using two dimensions:

- The first dimension represents cognitive evaluations of the company, which is **the company's competence (COMP)**.
- the second dimension captures affective judgments, which determine **the company's likeability (LIKE)**.

Exercise: Corporate Reputation

Objectives Definition



Exercise: Corporate Reputation

- Objectives Definition
- Data Gathering & Integration
- Data Exploration & Data Preparation
- Model Development & Model Evaluation
- Prediction and/or Interpretation

Objective: explain the effects of **corporate reputation** on **customer satisfaction** (CUSA) and, ultimately, **customer loyalty** (CUSL).

Data collection: a survey was conducted

Exercise: Corporate Reputation

Competence (*COMP*)	
comp_1	[The company] is a top competitor in its market
comp_2	As far as I know, [the company] is recognized worldwide
comp_3	I believe [the company] performs at a premium level
Likeability (*LIKE*)	
like_1	[The company] is a company I can better identify with than other companies
like_2	[The company] is a company I would regret more not having if it no longer existed than I would other companies
like_3	I regard [the company] as a likeable company
Customer satisfaction (*CUSA*)	
cusa	I am satisfied with [the company]
Customer loyalty (*CUSL*)	
cusl_1	I would recommend [company] to friends and relatives
cusl_2	If I had to choose again, I would choose [company] as my mobile phone service provider
cusl_3	I will remain a customer of [company] in the future

Source: Hair et al. (2022), Chap. 2; used with permission by Sage

Exercise: Corporate Reputation

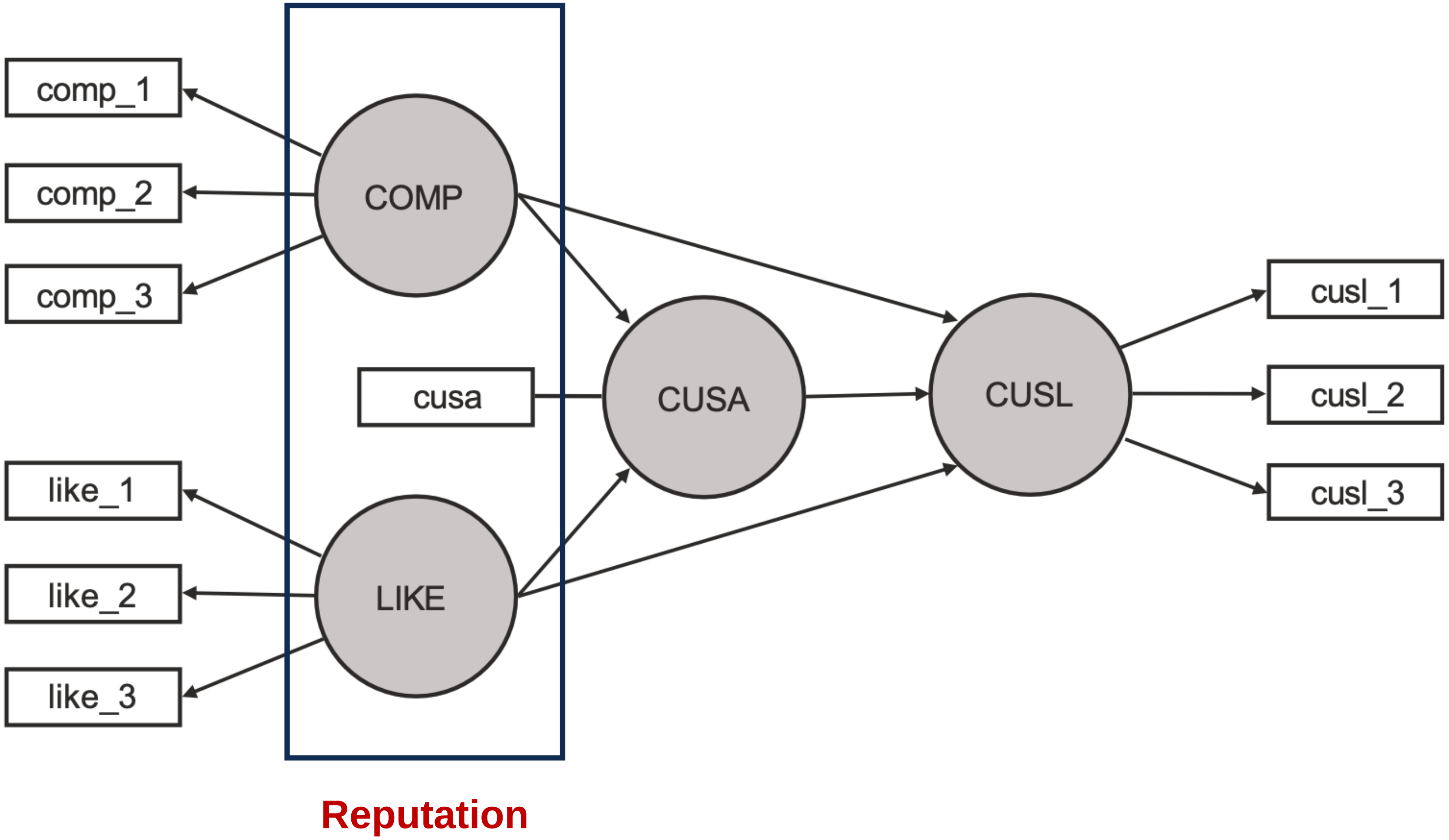
344 observations

1	serviceprovi	servicet	comp_1	comp_2	comp_3	like_1	like_2	like_3	cusl_1	cusl_2	cusl_3	cusa	csor_1	csor_2	csor_3	csor_4	csor_5	c
2	3	2	4	5	5	3	1	2	5	3	3	5	3	3	3	3	3	
3	3	2	6	7	6	6	6	6	7	7	7	7	2	5	6	4	6	
4	3	2	4	5	2	5	5	5	7	7	5	6	3	1	2	2	4	
5	3	2	6	4	4	6	5	6	7	7	7	6	3	3	5	3	5	
6	3	2	6	4	6	6	6	7	6	7	7	6	4	3	4	4	4	
7	3	2	3	4	4	6	7	7	7	7	7	6	3	3	4	3	3	
8	1	1	7	5	7	4	1	7	7	7	7	7	7	5	7	3	3	
9	1	1	6	6	6	4	3	4	5	4	6	4	4	1	3	3	2	
10	3	1	5	7	6	7	5	7	5	7	7	6	7	5	6	4	6	
11	3	2	6	5	5	6	6	6	6	6	7	6	4	1	5	2	4	
12	1	1	4	4	4	4	4	4	4	2	1	3	4	6	4	4	4	
13	1	1	3	6	2	4	6	5	4	5	6	4	4	3	4	4	3	
14	2	2	3	3	4	2	4	4	4	5	5	4	4	2	3	2	2	
15	1	2	5	7	7	3	4	7	7	7	7	5	4	4	4	4	3	
16	1	2	3	7	7	6	4	7	4	1	1	5	5	3	5	4	1	
17	1	2	3	3	3	3	2	3	4	4	4	4	3	2	3	1	1	

Exercise: Corporate Reputation

Objectives Definition	**Objective:** explain the effects of **corporate reputation** on **customer satisfaction** (CUSA) and, ultimately, **customer loyalty** (CUSL).
Data Gathering & Integration	**Data collection:** a survey was conducted
Data Exploration & Data Preparation	**Data exploration:** descriptive analysis was performed **Data preparation:** anomalies and inconsistencies have been identified and removed.
Model Development & Model Evaluation	PLS-SEM
Prediction and/or Interpretation	Interpretation

Exercise: Corporate Reputation



PLS-SEM: measurement model and structural model

Measurement Model Assessment

1. Assess the **indicator reliability** (loadings)
2. Assess the **internal consistency reliability** (Composite reliability rhoc and Cronbach's alpha)
3. Assess the **convergent validity** (AVE)
4. Assess the **discriminant validity** (HTMT)

Structural Model Assessment

1. Assess **collinearity issues** the structural model
2. Assess the **significance and relevance** of the structural model relationships
3. Assess the model's **explanatory power**

Preliminary Operations

```
df=read.csv2("/Users/gp/Desktop/06_Exercise 3 dataset.csv")

install.packages("seminr")
library("seminr")
summary(df)
```

Create the Measurement Model

SEMinR uses *the constructs()* function to specify the list of all construct measurement models. Within this list, we can then define various constructs. *composite()* specifies the measurement of individual constructs.

construct_name item_names weights

```
mm <- constructs(
    composite("COMP", multi_items("comp_", 1:3)),
    composite("LIKE", multi_items("like_", 1:3)),
    composite("CUSA", single_item("cusa")),
    composite("CUSL", multi_items("cusl_", 1:3))
        )
```

Create the Structural Model

SEMinR makes structural model specification more human readable, domain relevant, and explicit by using these functions:

- *relationships()* specifies all the **structural** relationships between all constructs.
- *paths()* specifies relationships between sets of antecedents and outcomes.

```
sm <- relationships(
   paths(from = c("COMP", "LIKE"), to = c("CUSA", "CUSL")),
paths(from = c("CUSA"), to = c("CUSL"))
)
sm
```

Estimating the Model

model <- estimate_pls(
data = df, ← The dataset containing the indicator
measurement_model = mm, ← The measurement model
structural_model = sm, ← The structural model
)

```
Generating the seminr model
All 344 observations are valid.
```

 # Summarize the model results
sum_model <- summary(model)
sum_model

Estimating the Model: Overview

```
Path Coefficients:
          CUSA   CUSL
R^2      0.015  0.380
AdjR^2   0.010  0.375
COMP     0.148  0.033
LIKE    -0.047  0.033
CUSA         .  0.608
```

Low Explanatory Power

Unrelevant Path Coefficient

Internal consistency reliability

```
Reliability:
     alpha  rhoC   AVE  rhoA
COMP 0.776 0.870 0.692 0.784
LIKE 0.831 0.897 0.744 0.878
CUSA 1.000 1.000 1.000 1.000
CUSL 0.421 0.692 0.530 0.755
```

Convergent validity

Internal consistency!

R-squared between 0.10 and 0.50

Cronbach's alpha between 0.70 and 0.90

Composite reliability between 0.70 and 0.90

AVE > 0,5

Estimating the Model: Indicator Reliability

sum_model$loadings

	COMP	LIKE	CUSA	CUSL
comp_1	0.766	0.000	0.000	0.000
comp_2	0.858	0.000	0.000	0.000
comp_3	0.868	0.000	0.000	0.000
like_1	0.000	0.903	0.000	0.000
like_2	0.000	0.859	0.000	0.000
like_3	0.000	0.824	0.000	0.000
cusa	0.000	0.000	1.000	0.000
cusl_1	0.000	0.000	-0.000	-0.002
cusl_2	0.000	0.000	0.000	0.874
cusl_3	0.000	0.000	0.000	0.909

Indicator reliability

Loadings > 0,7

Dealing with Indicator Reliability Issues

```
mm_2 <- constructs(
    composite("COMP", multi_items("comp_", 1:3)),
    composite("LIKE", multi_items("like_", 1:3)),
    composite("CUSA", single_item("cusa")),
    composite("CUSL", multi_items("cusl_", c(2,3)))
        )

model_2 <- estimate_pls(
data = df,
measurement_model = mm_2,
structural_model = sm,
)
```

Dealing with Indicator Reliability Issues

```
sum_model_2 <- summary(model_2)
sum_model_2$loadings
```

	COMP	LIKE	CUSA	CUSL
comp_1	0.766	0.000	0.000	0.000
comp_2	0.858	0.000	0.000	0.000
comp_3	0.868	0.000	0.000	0.000
like_1	0.000	0.903	0.000	0.000
like_2	0.000	0.859	0.000	0.000
like_3	0.000	0.824	0.000	0.000
cusa	0.000	0.000	1.000	0.000
cusl_2	0.000	0.000	0.000	0.874
cusl_3	0.000	0.000	0.000	0.909

Loadings > 0,7

Dealing with Indicator Reliability Issues

sum_model

```
Path Coefficients:
          CUSA  CUSL
R^2      0.015 0.380
AdjR^2   0.010 0.375
COMP     0.148 0.033
LIKE    -0.047 0.033
CUSA         . 0.608

Reliability:
     alpha  rhoC   AVE  rhoA
COMP 0.776 0.870 0.692 0.784
LIKE 0.831 0.897 0.744 0.878
CUSA 1.000 1.000 1.000 1.000
CUSL 0.421 0.692 0.530 0.755
```

Internal consistency!

sum_model_2

```
Path Coefficients:
          CUSA  CUSL
R^2      0.015 0.380
AdjR^2   0.010 0.375
COMP     0.148 0.033
LIKE    -0.047 0.033
CUSA         . 0.608

Reliability:
     alpha  rhoC   AVE  rhoA
COMP 0.776 0.870 0.692 0.784
LIKE 0.831 0.897 0.744 0.878
CUSA 1.000 1.000 1.000 1.000
CUSL 0.743 0.886 0.795 0.755
```

Estimating the Model: Collinearity Issues

sum_model_2$vif_antecedents

```
CUSA :
 COMP  LIKE
1.632 1.632

CUSL :
 COMP  LIKE  CUSA
1.655 1.635 1.016
```

VIF < 5

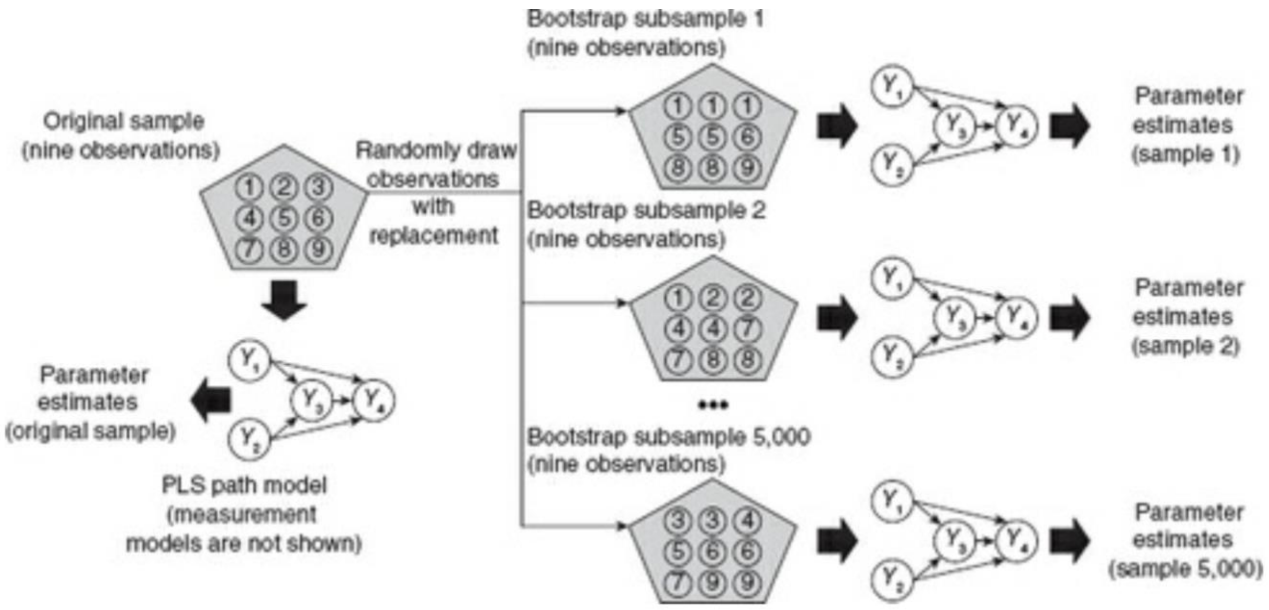
Estimating the Discriminant Validity: HTMT

sum_model_2$validity$htmt

```
      COMP  LIKE  CUSA CUSL
COMP     .     .     .    .
LIKE 0.780     .     .    .
CUSA 0.134 0.046     .    .
CUSL 0.164 0.102 0.708    .
>
```

HTMT < 0,9

Structural Model: Significance and Relevance of Path Coefficients



The bootstrap samples are used to estimate the PLS path model. That is, when using 5,000 bootstrap samples, 5,000 PLS path models are estimated.

The estimates of the coefficients form a **bootstrap distribution**, which can be viewed as an approximation of the sampling distribution. It is now possible to determine the **standard error** of the estimated coefficients.

(Hair et al., 2017)

Bootstrapping

```
# Bootstrap the model
boot_model_2<- bootstrap_model(
seminr_model = model_2,
nboot = 10000,
seed = 18
)

# Store the summary of the bootstrapped model
sum_boot_2<- summary(boot_model_2, alpha = 0.05)
```

5,000 is the standard, 10,000 are recommended!

Bootstrapping: Path Coefficients

 # Inspect the bootstrapped structural paths

sum_boot_2$bootstrapped_paths

	Original Est.	Bootstrap Mean	Bootstrap SD	T Stat.	2.5% CI	97.5% CI
COMP -> CUSA	0.148	0.167	0.052	2.855	0.058	0.274
COMP -> CUSL	0.033	0.036	0.043	0.777	-0.046	0.122
LIKE -> CUSA	-0.047	0.107	0.262	-0.180	-0.216	0.529
LIKE -> CUSL	0.033	0.047	0.080	0.411	-0.096	0.260
CUSA -> CUSL	0.608	0.524	0.288	2.114	0.104	0.943

Significance

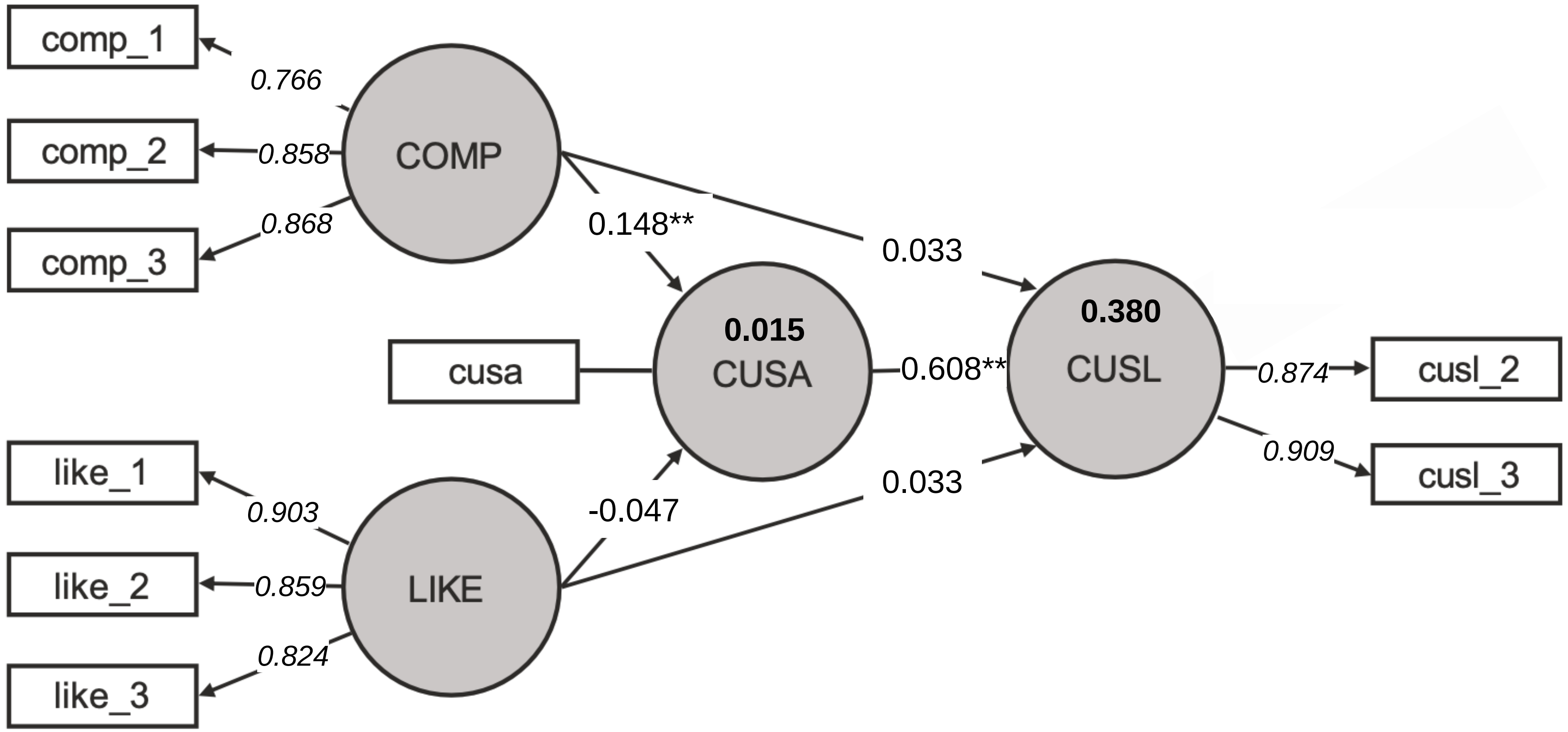
Bootstrapping: Path Coefficients

sum_boot_2<- summary(boot_model_2, alpha = 0.01)
sum_boot_2$bootstrapped_paths

	Original Est.	Bootstrap Mean	Bootstrap SD	T Stat.	0.5% CI	99.5% CI
COMP -> CUSA	0.148	0.167	0.052	2.855	0.023	0.310
COMP -> CUSL	0.033	0.036	0.043	0.777	-0.108	0.163
LIKE -> CUSA	-0.047	0.107	0.262	-0.180	-0.248	0.560
LIKE -> CUSL	0.033	0.047	0.080	0.411	-0.151	0.347
CUSA -> CUSL	0.608	0.524	0.288	2.114	0.075	0.981

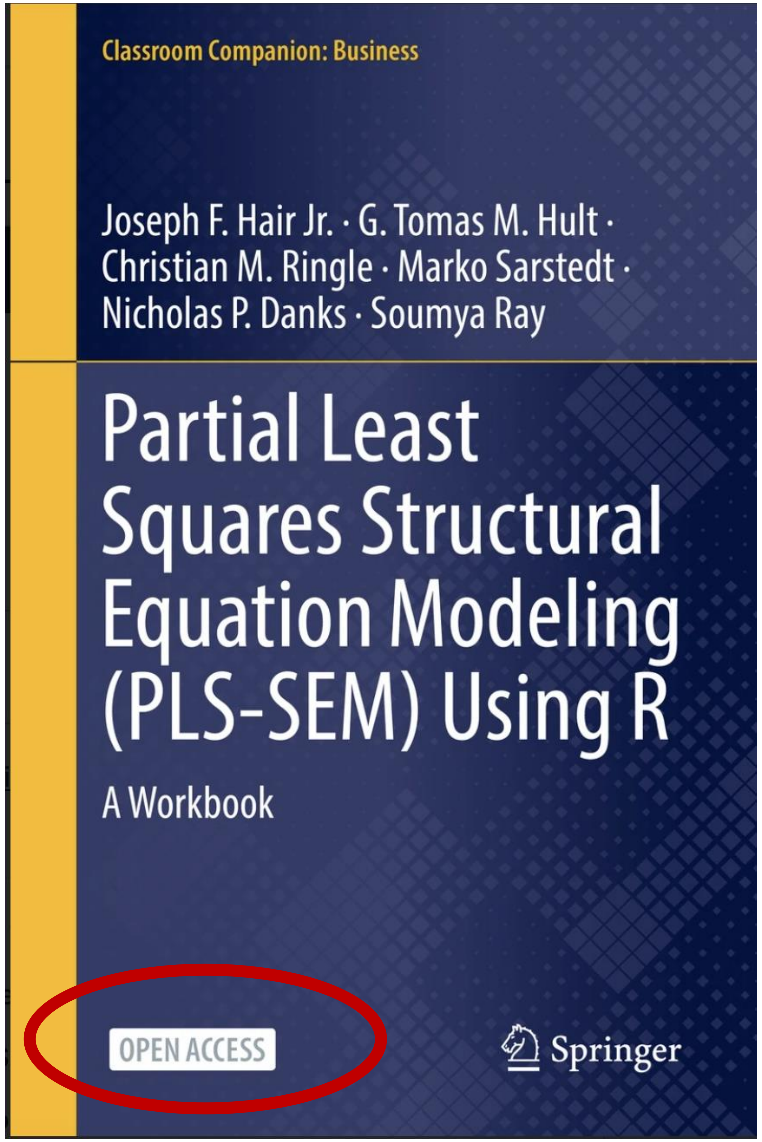
Significance

The tested model



** p<0.01

For R Users:



link.springer.com/book/10.1007/978-3-030-80519-7

POLITECNICO
MILANO 1863

Gloria Peggiani

AY 2024/2025