

# Use of the R package TextSEM

The R package TextSEM can be used for SEM analysis with text data. To install the package, please use

```
## Install the package for text analysis
remotes::install_github("Stan7s/TextSEM")

## The package can be installed from CRAN directly in the future
# install.packages('TextSEM')
```

We now illustrate the use of the package through several examples.

## Sentiment analysis

In this example, we introduce how to use the function `sem.sentiment` to extract sentiment variables from text and estimate the SEM model. Specifically, the overall sentiment of comment is extracted and used as a mediator between three endogenous variables (book, attendance, difficulty) and two exogenous variables (grade and rating).

To use this function, we need to first specify the model:

```
model <- ' rating ~ book + attendance + difficulty + comments
          grade ~ book + attendance + difficulty + comments
          comments ~ book + attendance + difficulty
          '
```

The function `sem.sentiment` requires three parameters: the structural equation model, the input data frame, and the name of the text variable in the data frame to be analyzed for sentiment.

```
res <- sem.sentiment(model = model,
                     data = prof1000,
                     text_var=c('comments'))
summary(res$estimates, fit = TRUE)
```

The output of the analysis is given below:

lavaan 0.6.17 ended normally after 63 iterations

Estimator	ML
Optimization method	NLMINB
Number of model parameters	27
Number of observations	38240
Number of missing patterns	8

Model Test User Model:

Test statistic	0.000
Degrees of freedom	0

Model Test Baseline Model:

Test statistic	31563.154
Degrees of freedom	12
P-value	0.000

User Model versus Baseline Model:

Comparative Fit Index (CFI)	1.000
Tucker-Lewis Index (TLI)	1.000
Robust Comparative Fit Index (CFI)	1.000
Robust Tucker-Lewis Index (TLI)	1.000

Loglikelihood and Information Criteria:

Loglikelihood user model (H0)	-160948.572
Loglikelihood unrestricted model (H1)	-160948.572
Akaike (AIC)	321951.144
Bayesian (BIC)	322182.038
Sample-size adjusted Bayesian (SABIC)	322096.232

Root Mean Square Error of Approximation:

RMSEA	0.000
90 Percent confidence interval - lower	0.000
90 Percent confidence interval - upper	0.000
P-value H_0: RMSEA <= 0.050	NA
P-value H_0: RMSEA >= 0.080	NA

Robust RMSEA	0.000
90 Percent confidence interval - lower	0.000
90 Percent confidence interval - upper	0.000
P-value H_0: Robust RMSEA <= 0.050	NA
P-value H_0: Robust RMSEA >= 0.080	NA

Standardized Root Mean Square Residual:

SRMR	0.000
------	-------

Parameter Estimates:

Standard errors	Standard
Information	Observed
Observed information based on	Hessian

Regressions:

	Estimate	Std.Err	z-value	P(> z )
rating ~				
book	0.169	0.013	12.905	0.000
attendance	0.127	0.023	5.618	0.000
difficulty	-0.331	0.004	-75.262	0.000
cmmnts.OvrllSn	2.671	0.021	125.974	0.000
grade ~				
book	-0.080	0.051	-1.558	0.119
attendance	-0.170	0.056	-3.058	0.002
difficulty	0.742	0.020	36.382	0.000
cmmnts.OvrllSn	-1.756	0.102	-17.171	0.000
comments.OverallSenti ~				
book	0.043	0.003	13.053	0.000
attendance	0.031	0.006	5.290	0.000
difficulty	-0.074	0.001	-73.666	0.000

Covariances:

	Estimate	Std.Err	z-value	P(> z )
.rating ~~				
.grade	-0.558	0.024	-23.191	0.000
book ~~				
attendance	0.017	0.002	8.374	0.000
difficulty	0.030	0.004	8.650	0.000
attendance ~~				
difficulty	0.028	0.006	4.712	0.000

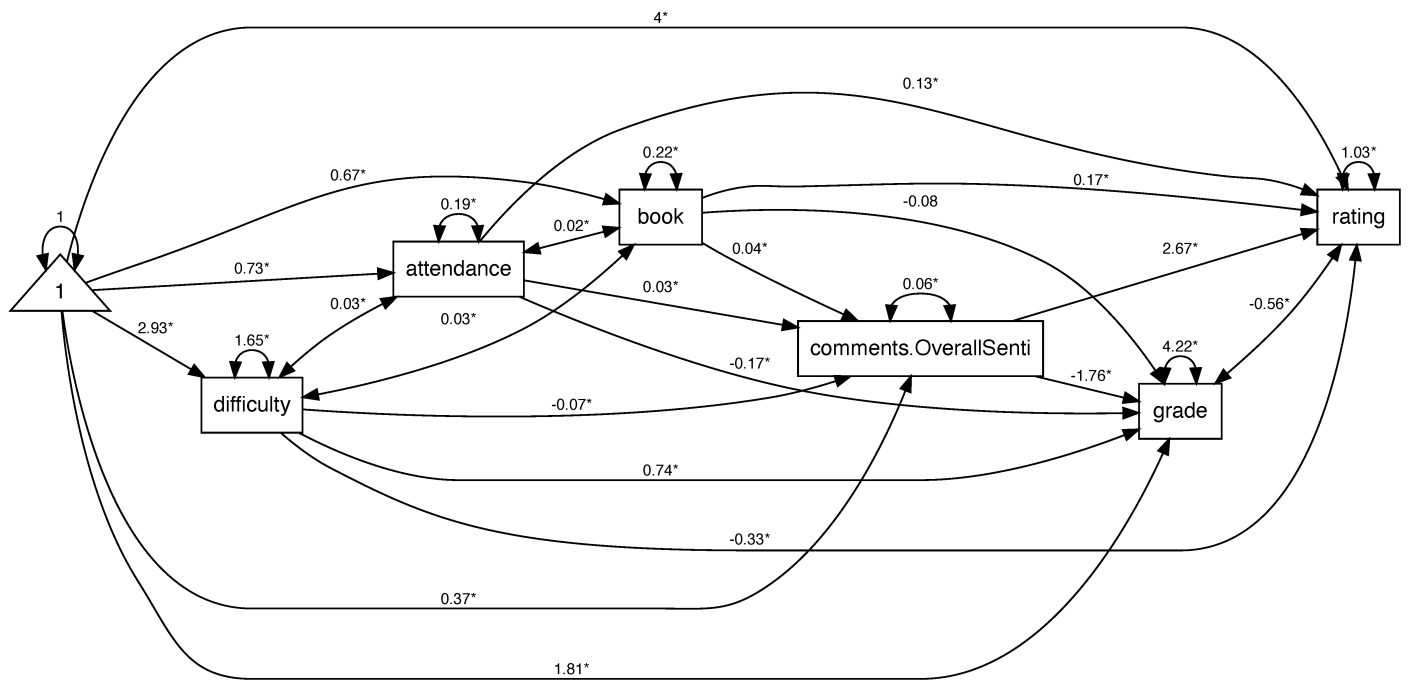
Intercepts:

	Estimate	Std.Err	z-value	P(> z )
.rating	3.995	0.022	182.815	0.000
.grade	1.807	0.084	21.559	0.000
.cmmnts.OvrllSn	0.367	0.005	69.455	0.000
book	0.673	0.003	245.275	0.000
attendance	0.732	0.004	164.946	0.000
difficulty	2.928	0.007	445.625	0.000

Variances:

	Estimate	Std.Err	z-value	P(> z )
.rating	1.034	0.008	136.885	0.000
.grade	4.222	0.069	61.443	0.000
.cmmnts.OvrllSn	0.061	0.000	136.529	0.000
book	0.220	0.002	120.633	0.000
attendance	0.195	0.003	71.124	0.000
difficulty	1.651	0.012	138.275	0.000

The path diagram for the model is



# Topic modeling

Students' comments about an instructor typically cover multiple topics, such as teaching style, classroom climate, and homework assignments. To identify these topics exploratorily and understand their relationships with other variables, we can apply the `sem.topic` function. This function performs topic modeling and estimates the SEM model including those identified topics.

In this example, we combine the comments from multiple students for each instructor. We also get the average scores for other variables.

```
prof.nest <- prof1000 %>% group_by(profid) %>%
  summarise(comments = paste(comments, collapse = " "),
    tags = paste(tags, collapse = ";"),
    rating = mean(rating, na.rm = TRUE),
    difficulty=mean(difficulty, na.rm = TRUE),
    book = mean(book, na.rm = TRUE),
    grade=mean(grade, na.rm = TRUE))
```

In addition to the three required parameters for `sem.sentiment` – model, data, and text variables, the `sem.topic` function requires an additional parameter: `n_topics`. This parameter specifies the number of topics to extract from each column of the text data. Based on previous cross-validation analysis (Jacobucci et al., 2023), six topics were identified in this dataset. Consequently, we will extract six topics. Note that only the first  $n - 1$  topics will be incorporated into the SEM to avoid perfect multicollinearity, where  $n$  is the total number of topics specified.

```

model <- ' rating ~ book + difficulty + comments'
res <- sem.topic(model = model,
  data = prof.nest,
  text_var = c('comments'),
  n_topics = c(6))
summary(res$estimates, fit=TRUE)

```

The output is given below:

lavaan 0.6.17 ended normally after 1 iteration

Estimator	ML
Optimization method	NLMINB
Number of model parameters	8
	Used    Total
Number of observations	984    1000

Model Test User Model:

Test statistic	0.000
Degrees of freedom	0

Model Test Baseline Model:

Test statistic	1143.062
Degrees of freedom	7
P-value	0.000

User Model versus Baseline Model:

Comparative Fit Index (CFI)	1.000
Tucker-Lewis Index (TLI)	1.000

Loglikelihood and Information Criteria:

Loglikelihood user model (H0)	-631.624
Loglikelihood unrestricted model (H1)	-631.624
Akaike (AIC)	1279.248

Bayesian (BIC)	1318.381
Sample-size adjusted Bayesian (SABIC)	1292.973

Root Mean Square Error of Approximation:

RMSEA	0.000
90 Percent confidence interval - lower	0.000
90 Percent confidence interval - upper	0.000
P-value H <sub>0</sub> : RMSEA ≤ 0.050	NA
P-value H <sub>0</sub> : RMSEA ≥ 0.080	NA

Standardized Root Mean Square Residual:

SRMR	0.000
------	-------

Parameter Estimates:

Standard errors	Standard
Information	Expected
Information saturated (h1) model	Structured

Regressions:

	Estimate	Std.Err	z-value	P(> z )
rating ~				
book	0.295	0.058	5.094	0.000
difficulty	-0.335	0.023	-14.663	0.000
comments.topc1	0.392	0.106	3.696	0.000
comments.topc2	2.503	0.102	24.531	0.000
comments.topc3	1.637	0.105	15.554	0.000
comments.topc4	-0.344	0.090	-3.799	0.000
comments.topc5	0.273	0.093	2.955	0.003

Variances:

	Estimate	Std.Err	z-value	P(> z )
.rating	0.211	0.010	22.181	0.000

## Text embedding

Embedding techniques offer an advantage over topic models in their ability to construct latent factors in higher dimensions from textual data. In this example, we demonstrate how to leverage embedding techniques within the framework of SEM using the `sem.emb` function.

Before we start, we need to set up the Python environment with the `reticulate` package, which provides a bridge between R and Python. The code below can be used for the purpose.

```
library(reticulate)

## First time set-up
virtualenv_create("r-reticulate")
py_install("transformers")
py_install("torch")
py_install("sentence_transformers")
py_install("openai")

## Call virtual environment
use_virtualenv("r-reticulate")
```

Although it is not required, we recommended first to embed the text and then include the embedded vectors in the SEM analysis. The reason is that text embedding can be time consuming. The embedded data can also be used in multiple models rather than just the model specified.

We can use the `sem.encode` function to generate text embeddings. This function supports pre-trained models from SentenceBERT and OpenAI. Here, we'll use the all-mpnet-base-v2 model from SentenceBERT. Note that when using OpenAI models, an API key must be specified in the system directory.

```
embeddings <- sem.encode(prof.nest$comments,
                        encoder = "all-mpnet-base-v2")

## save the embeddings
save(embeddings, file="data/prof.nest.emb.rda")
```

We then incorporate these embeddings into an SEM model using the `sem.emb` function. This function allows us to integrate the rich semantic information captured by the embeddings into our statistical model. Two key parameters in this function are: 1) `pca_dim`: the number of dimensions to retain after applying PCA to the embeddings, and 2) `emb_filepath`: the file path to the saved embeddings.

```
sem_model <- ' rating ~ book + difficulty + comments'
res <- sem.emb(sem_model = sem_model,
              data = prof.nest,
              text_var = "comments",
```



```
pca_dim = 10,  
emb_filepath = "data/prof.nest.emb.rda")
```

The output looks like:

lavaan 0.6.17 ended normally after 1 iteration

Estimator	ML	
Optimization method	NLMINB	
Number of model parameters	12	
	Used	Total
Number of observations	984	1000

Model Test User Model:

Test statistic	0.000
Degrees of freedom	0

Model Test Baseline Model:

Test statistic	887.411
Degrees of freedom	11
P-value	0.000

User Model versus Baseline Model:

Comparative Fit Index (CFI)	1.000
Tucker-Lewis Index (TLI)	1.000

Loglikelihood and Information Criteria:

Loglikelihood user model (H0)	-759.449
Loglikelihood unrestricted model (H1)	-759.449
Akaike (AIC)	1542.898
Bayesian (BIC)	1601.598
Sample-size adjusted Bayesian (SABIC)	1563.486

Root Mean Square Error of Approximation:

RMSEA	0.000
90 Percent confidence interval - lower	0.000
90 Percent confidence interval - upper	0.000
P-value H <sub>0</sub> : RMSEA ≤ 0.050	NA
P-value H <sub>0</sub> : RMSEA ≥ 0.080	NA

Standardized Root Mean Square Residual:

SRMR	0.000
------	-------

Parameter Estimates:

Standard errors	Standard
Information	Expected
Information saturated (h1) model	Structured

Regressions:

	Estimate	Std.Err	z-value	P(> z )
rating ~				
book	0.168	0.067	2.517	0.012
difficulty	-0.406	0.026	-15.524	0.000
comments.PC1	-10.239	0.549	-18.654	0.000
comments.PC2	-4.308	0.539	-7.998	0.000
comments.PC3	7.982	0.573	13.931	0.000
comments.PC4	-1.373	0.526	-2.612	0.009
comments.PC5	-0.484	0.534	-0.906	0.365
comments.PC6	0.034	0.531	0.064	0.949
comments.PC7	2.664	0.531	5.019	0.000
comments.PC8	-1.183	0.527	-2.243	0.025
comments.PC9	0.408	0.531	0.767	0.443

Variances:

	Estimate	Std.Err	z-value	P(> z )
.rating	0.274	0.012	22.181	0.000

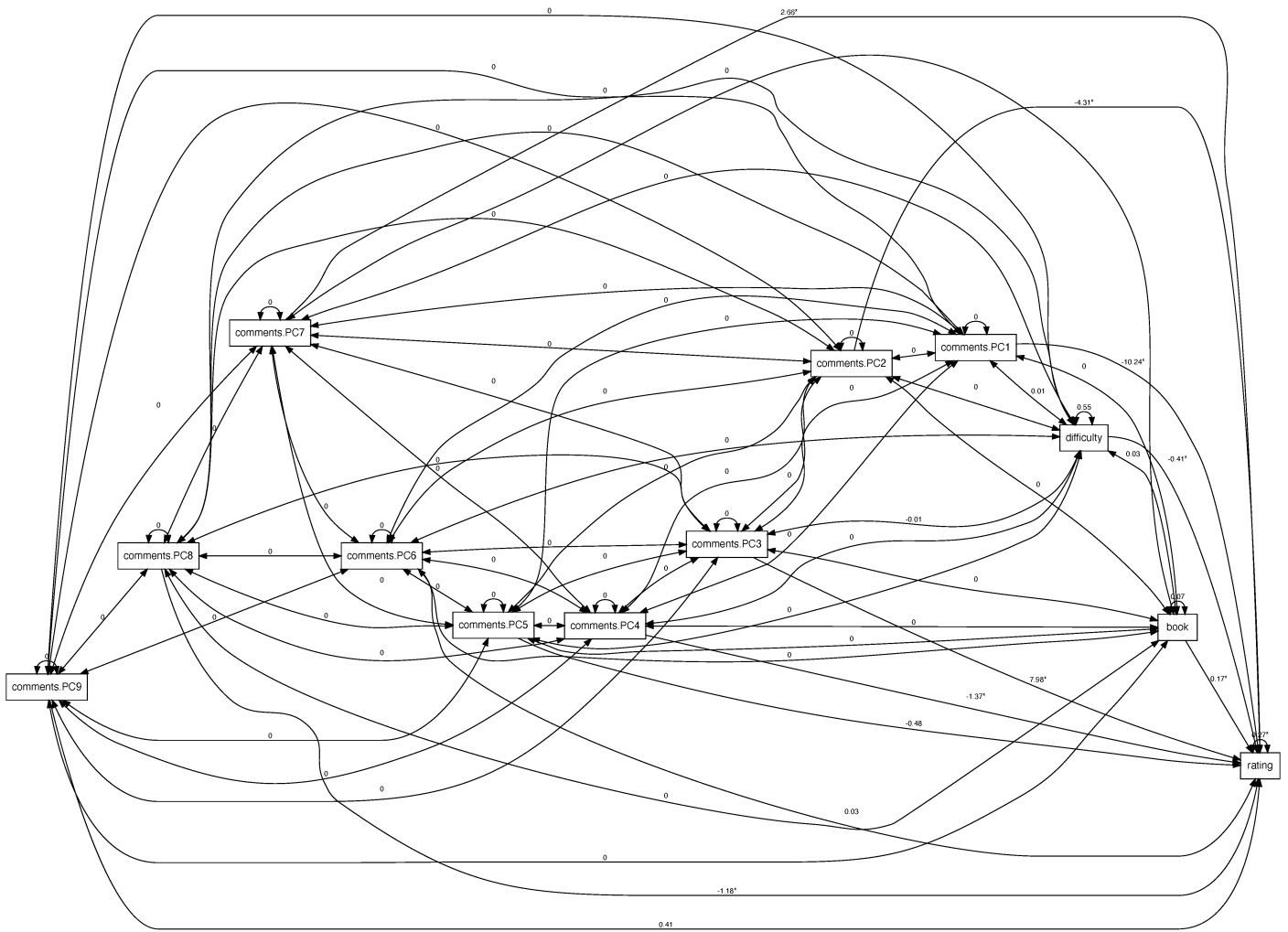
Note that to embed the text and conduct the analysis at the same time, one can use

```
res <- sem.emb(sem_model = sem_model,
              data = prof.nest,
```

```

text_var = "comments",
pca_dim = 10,
encoder = "all-mpnet-base-v2")

```



Revision #3

Created 24 October 2024 20:58:15 by Admin

Updated 25 October 2024 14:28:36 by Admin