Statistics for Linguistics

Session 6

Linear Mixed Effects Regression Models

Part 1 – Pre-Processing

Let‘s take a step back

Pre-Processing

- As the number of variables and the complexity of models increases, one has to consider a number of important factors

1. Variable distribution
2. Assumptions of linear models

Pre-Processing

- As the number of variables and the complexity of models increases, one has to consider a number of important factors

1. Variable distribution ✓
2. Assumptions of linear models ✓

Pre-Processing

- As the number of variables and the complexity of models increases, one has to consider a number of important factors

1. Variable distribution ✓
2. Assumptions of linear models ✓
3. Collinearity

Collinearity

- **Collinearity** is a phenomenon in which one predictor variable in a multiple regression model can be linearly predicted from the others with a substantial degree of accuracy
- In this situation, the **coefficient estimates** of the multiple regression may **change erratically** in response to small changes in the model or the data
- It only affects calculations regarding **individual predictors**:
  - That is, a multivariate regression model with collinear predictors can indicate how well the entire bundle of predictors predicts the outcome variable, but it may not give valid results about any individual predictor, or about which predictors are redundant with respect to others

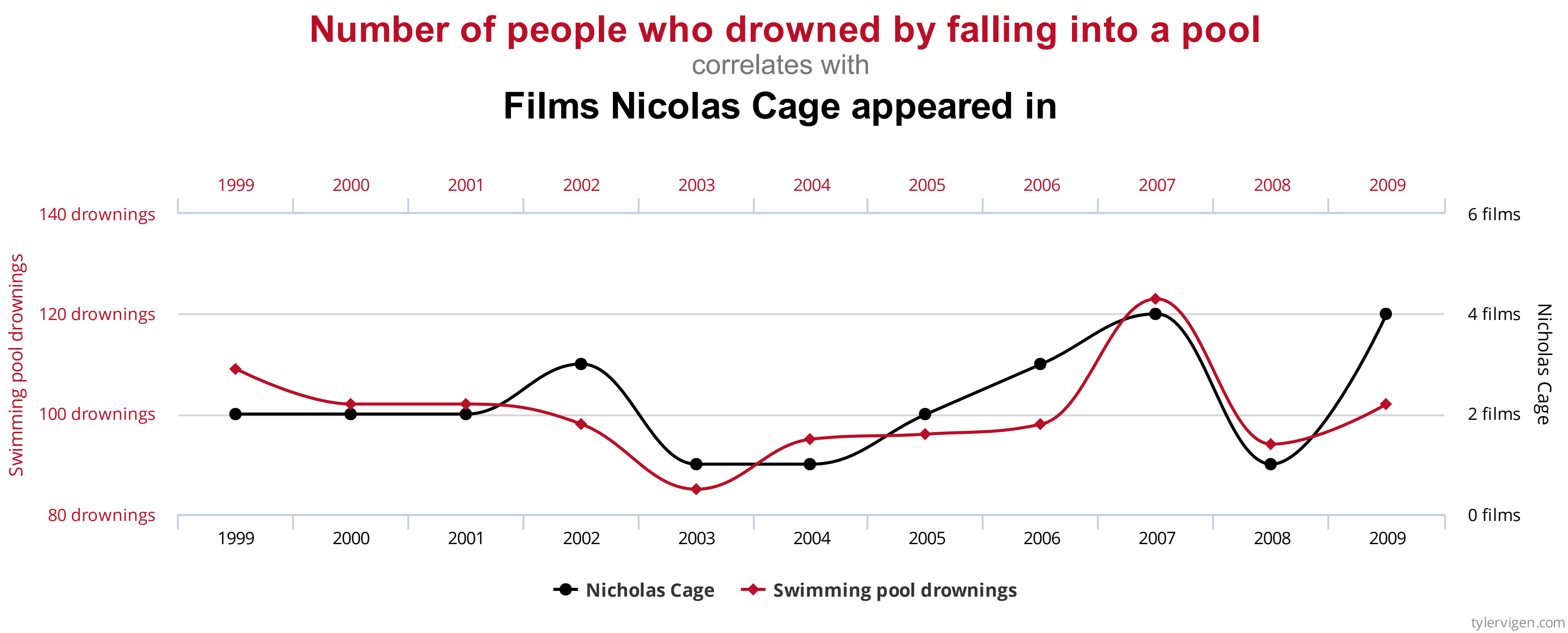
Collinearity

- **Collinearity** is brought into a model by **correlated** predictor variables
- The more predictor variables we introduce to a model, the higher the chance of correlated variables becomes
- **Correlation**

> "Correlation is any statistical relationship, whether causal or not, between two variables. In the broadest sense, correlation is any statistical association, though it commonly refers to the degree to which a pair of variables are linearly related."

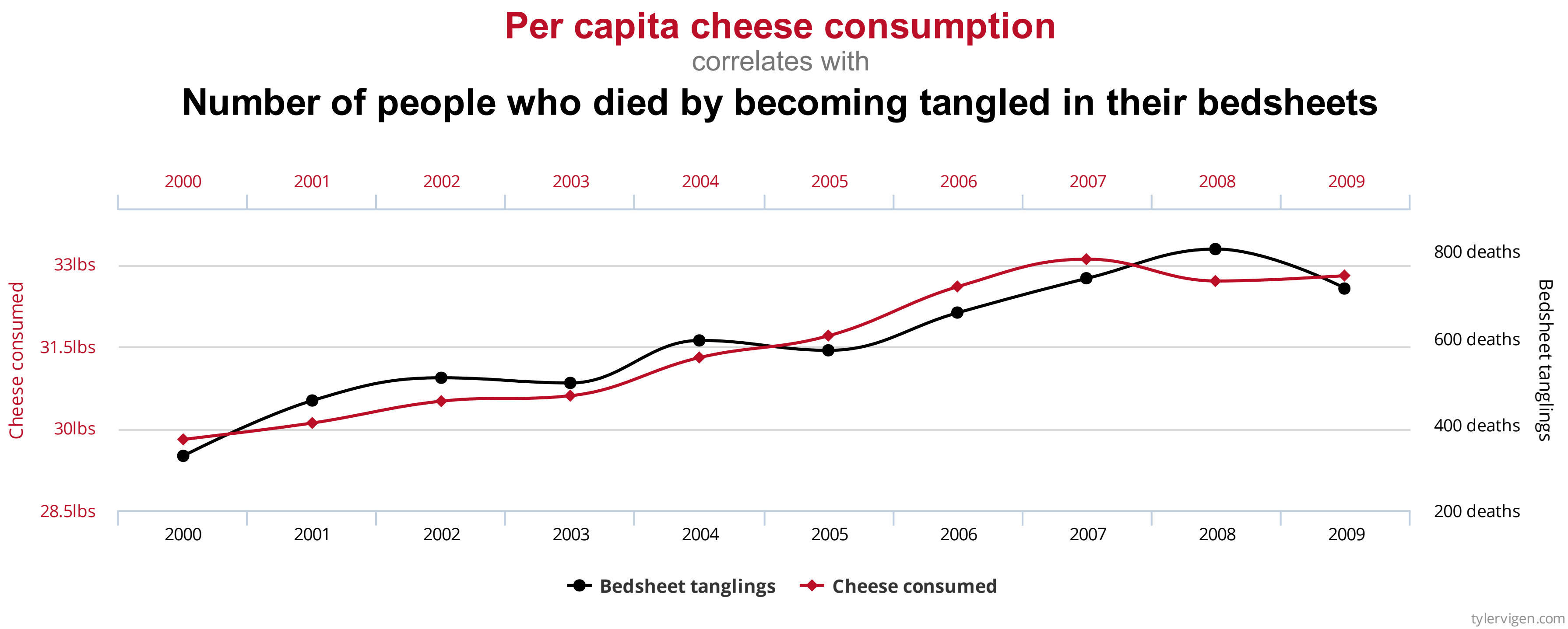
Correlation

- Important: correlation ≠ causation



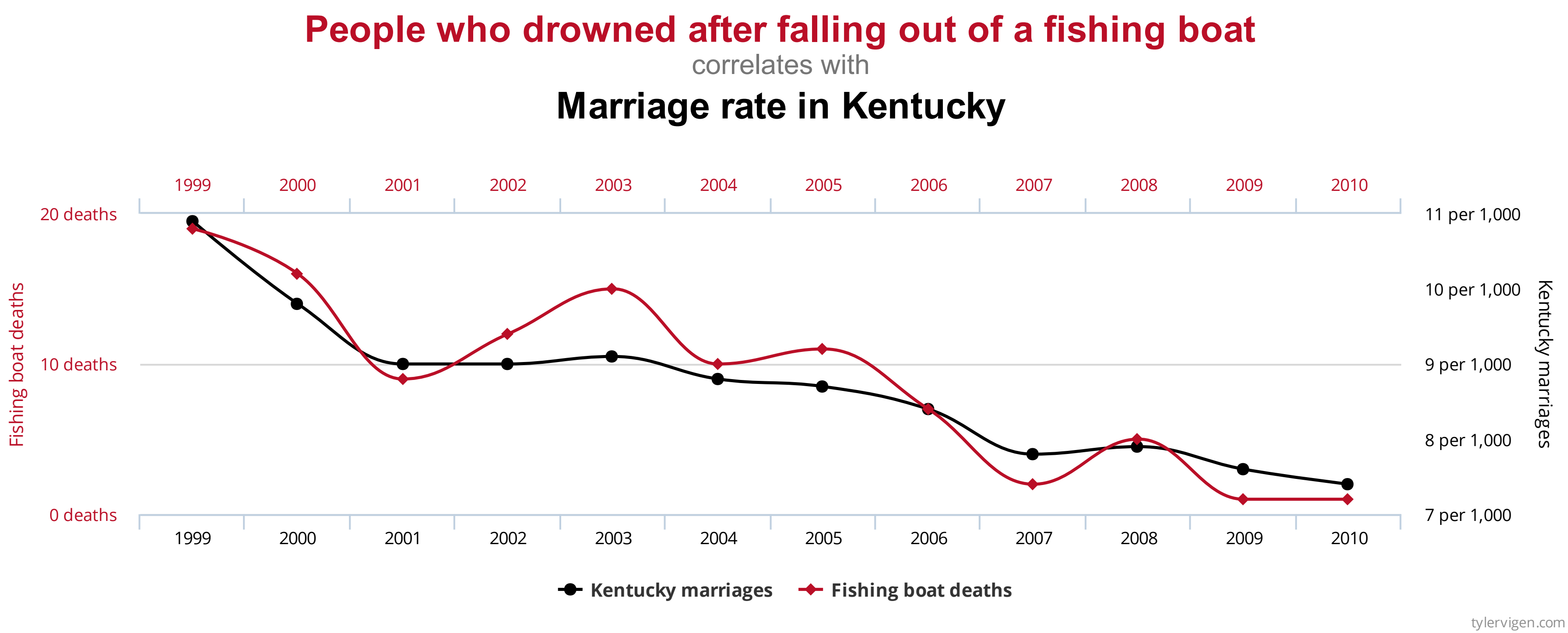
Correlation

- Important: correlation ≠ causation



Correlation

- Important: correlation ≠ causation



Correlation

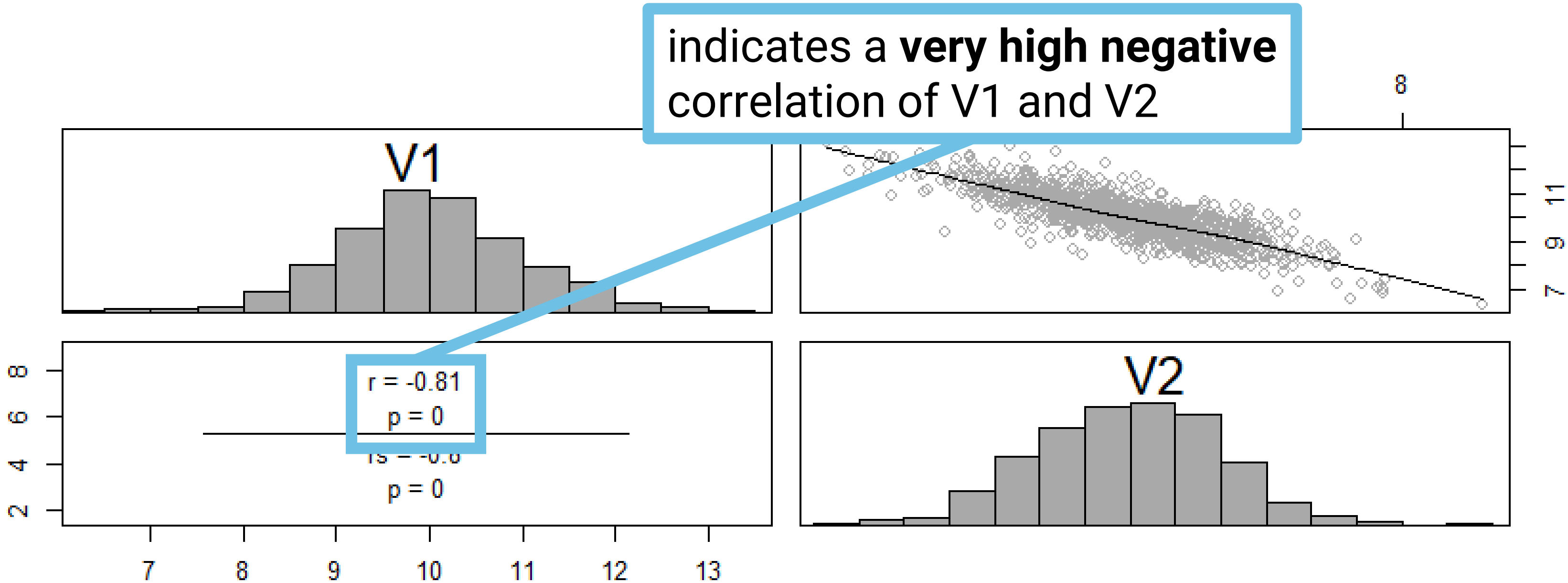
- Correlation can be measured using
  - **Pearson's r** for numerical variables
  - **Spearman's** rho for all other comparisons

correlation coefficient			label	sign
0.7	$< r \leq$	1	very high	positive correlation
0.5	$< r \leq$	0.7	high	
0.2	$< r \leq$	0.5	intermediate	
0	$< r \leq$	0.2	low	
$r \approx 0$			no statistical correlation	
0	$> r \geq$	-0.2	low	negative correlation
-0.2	$> r \geq$	-0.5	intermediate	
-0.5	$> r \geq$	-0.7	high	
-0.7	$> r \geq$	-1	very high	

Correlation

- The `SfL` package provides a function to compute correlation coefficients by creating a so-called correlation matrix plot

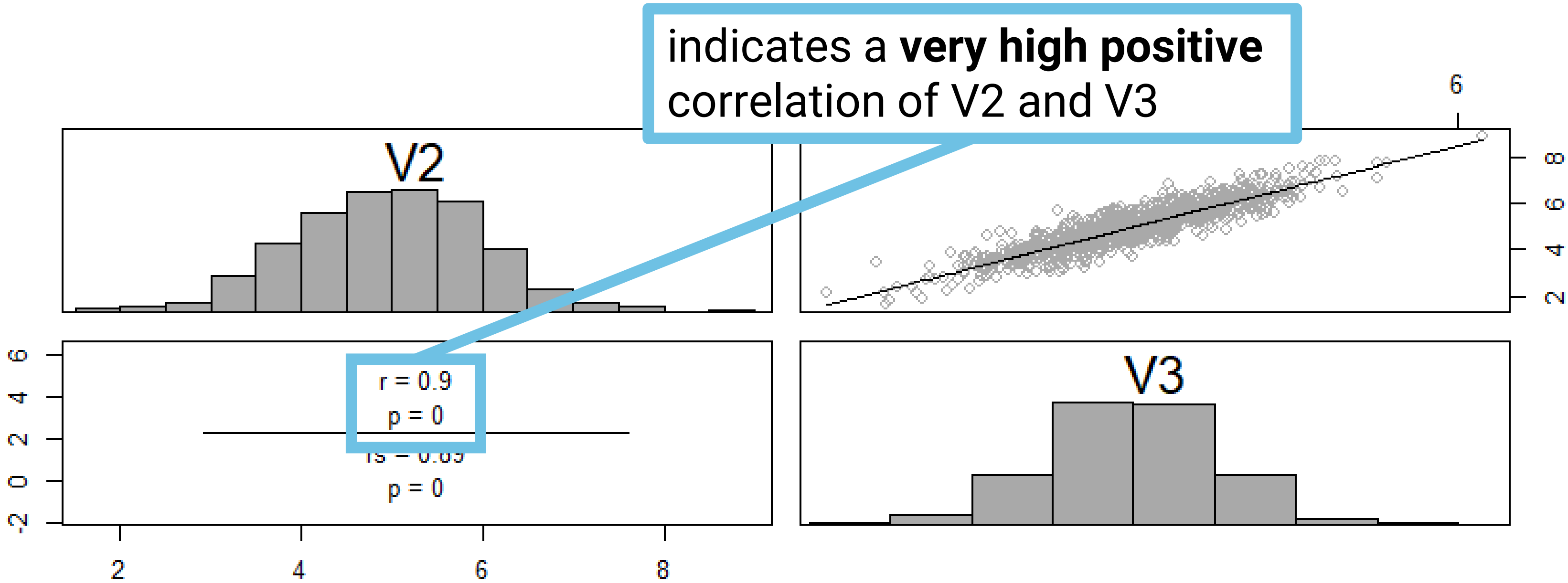
```
correlation_matrix(data = df, variables = c("V1", "V2"))
```



Correlation

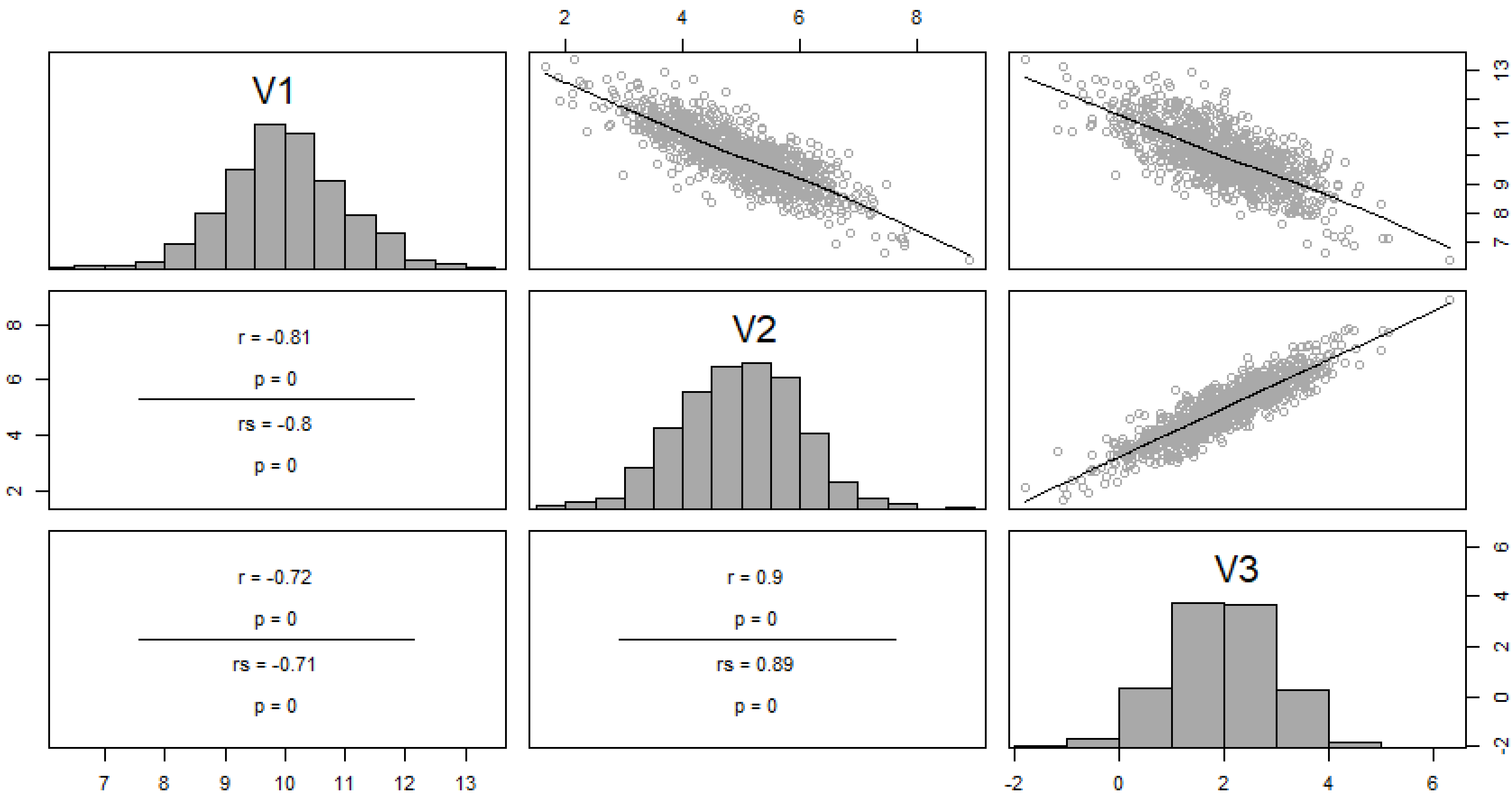
- The `SfL` package provides a function to compute correlation coefficients by creating a so-called correlation matrix plot

```
correlation_matrix(data = df, variables = c("V2", "V3"))
```



Correlation

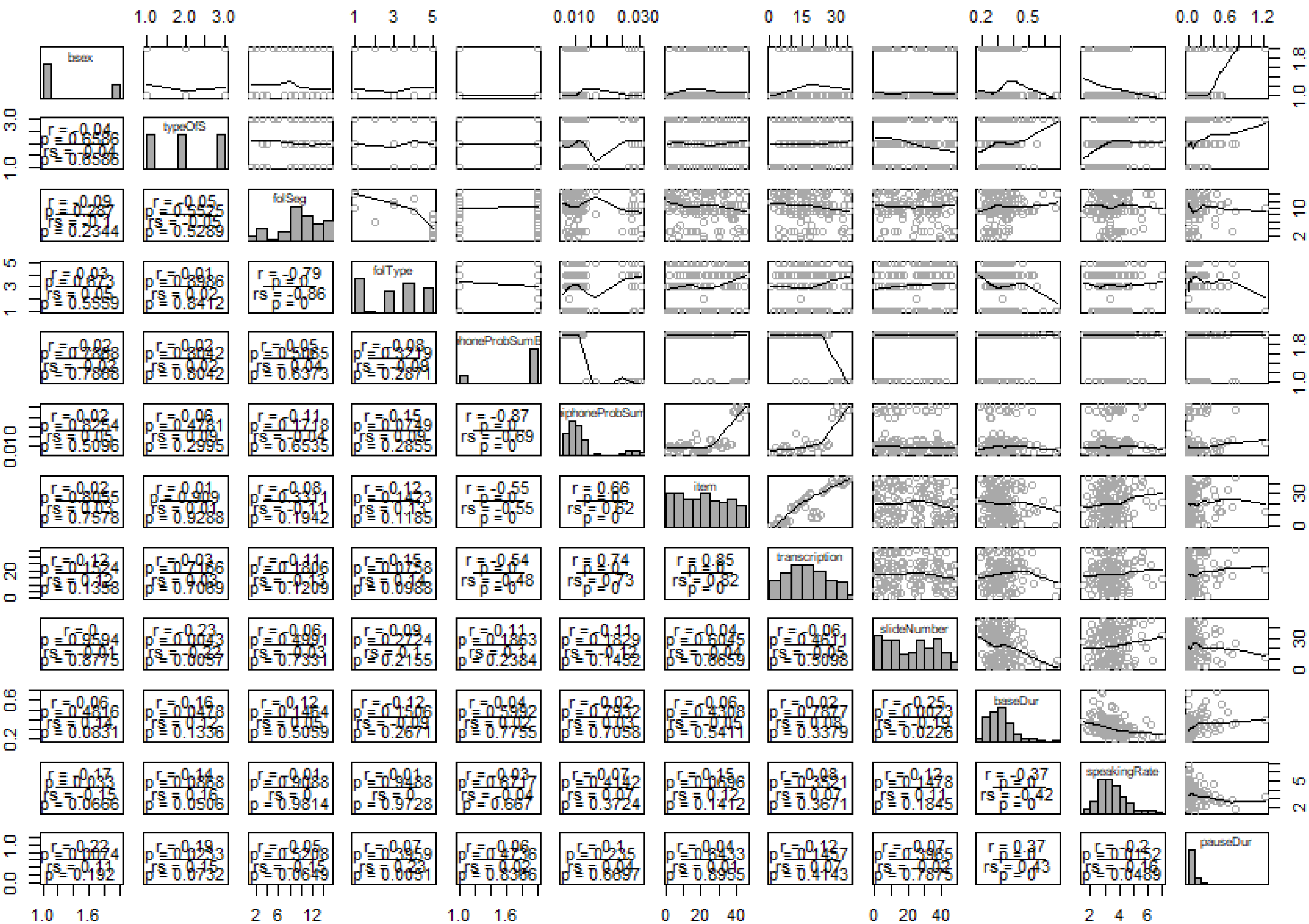
```
correlation_matrix(data = df, variables = c("V1", "V2", "V3"))
```



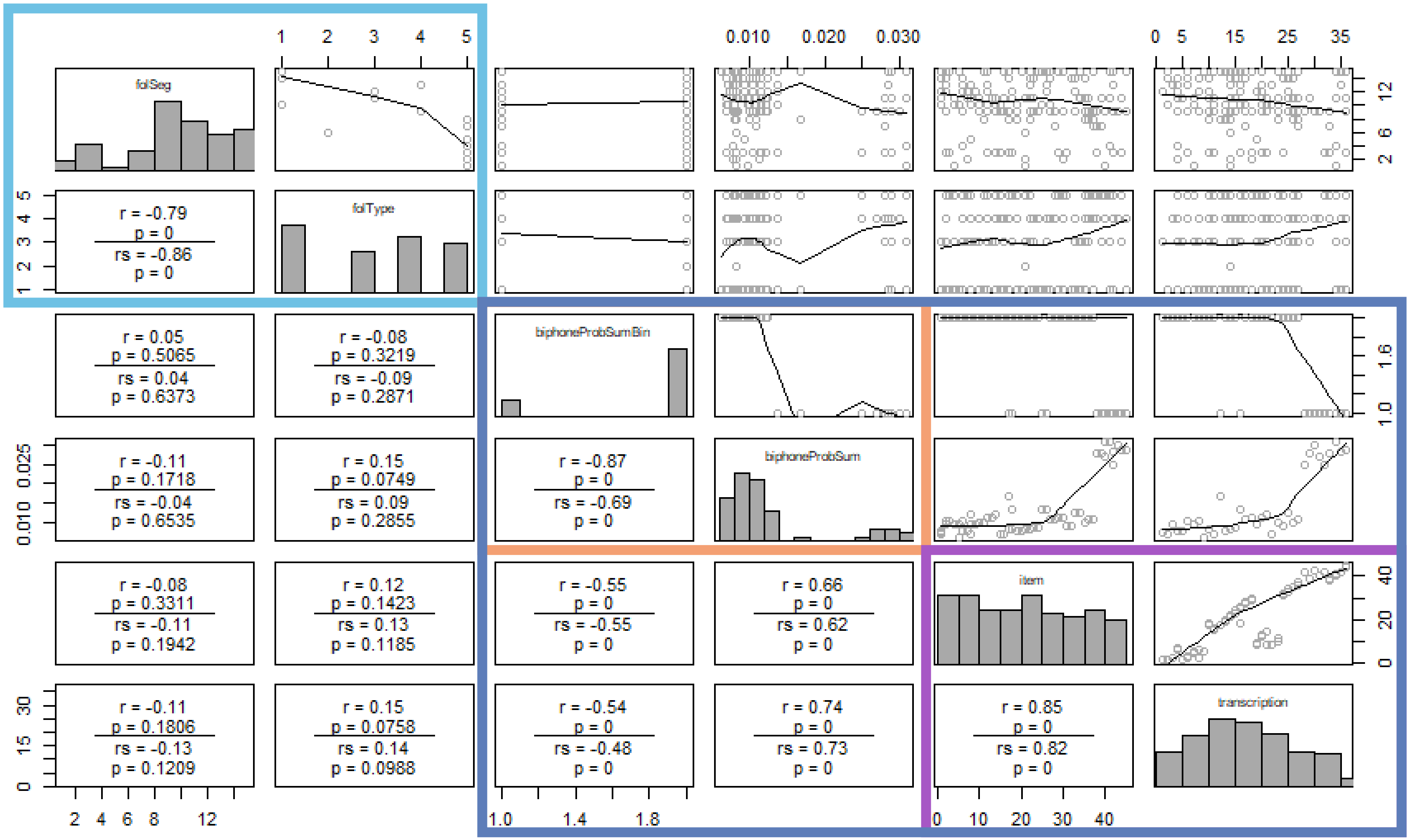
Collinearity / Correlation

- Of course, so far this is only simulated data
- What is the situation for real data, e.g. for the word-final /s/ data set?

Correlation in data_s



Correlation in data_s



Correlation in data_s

var1	var2	r / rho
folSeg	folType	-0.86
biphoneProbSumBin	biphoneProbSum	-0.69
biphoneProbSumBin	item	-0.55
biphoneProbSumBin	transcription	-0.48
biphoneProbSum	item	0.62
biphoneProbSum	transcription	0.73
item	transcription	0.82

Correlation in data_s

var1	var2	r / rho
folSeg	folType	-0.86
biphoneProbSumBin	**biphoneProbSum**	-0.69
biphoneProbSumBin	**item**	-0.55
biphoneProbSumBin	**transcription**	-0.48
biphoneProbSum	**item**	0.62
biphoneProbSum	**transcription**	0.73
item	**transcription**	0.82

Strategies for Collinearity Issues

- But what to do if we find highly correlated variables?
- **Strategy 1**

  Find out which variable out of a pair of highly correlated variables is the better predictor. Retain this better predictor, get rid off the other variable.

  → this works for all types of predictor variables
- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

  → this works best for numeric predictor variables

Strategy 1: data_s

- **Strategy 1**

  Find out which variable out of a pair of highly correlated variables is the better predictor.

  Retain this better predictor, get rid off the other variable.

- **Step 1**

  Create a simple model for all variables that are highly correlated with other variables, e.g.

```
mdl_item <- lm(sDurLog ~ item, data_s)

mdl_transcription <- lm(sDurLog ~ transcription, data_s)
```

Strategy 1: data_s

- **Strategy 1**

  Find out which variable out of a pair of highly correlated variables is the better predictor.

  Retain this better predictor, get rid off the other variable.

- **Step 2**

  Compare the simple models of highly correlated with each other to find the better predictor, e.g.

```
> anova(mdl_item, mdl_transcription)
Analysis of Variance Table
Model 1: sDurLog ~ item
Model 2: sDurLog ~ transcription
  Res.Df    RSS Df Sum of Sq      F  Pr(>F)
1    105 16.919
2    114 19.843 -9   -2.9244 2.0165 0.04442 *
```

Residual sum-of-squares
The lower, the better the model fit

Strategy 1: data_s

- **Strategy 1**

  Find out which variable out of a pair of highly correlated variables is the better predictor.

  Retain this better predictor, get rid off the other variable.

- **Step 3**

  Keep the predictor of the better fit model, i.e. in this case: `item`

  Ignore the other variable (i.e. `transcription`) for the reminder of the analysis

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 1**

  Create a data frame only consisting of the highly correlated variables. In our simulated data, the following variables are highly correlated with at least one other variable:

  `var1, var2, var3, var4, var5, var6, var7, var8, var9`

- This is basically the worst case scenario of collinearity: all variables show high correlation with at least one other variable

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 2**

  Use the principal component function to compute a principle component analysis:

```
pc <- princomp(df, cor=TRUE, score=TRUE)
```

Strategy 2: simulated data

- **Strategy 2**
  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.
- **Step 3**
  Take a closer look at the **Eigenvalues** of the newly created variables using the `factoextra` package:

```
> library("factoextra")
> get_eigenvalue(pc)
       eigenvalue variance.percent cumulative.variance.percent
Dim.1 2.71706780       30.1896422                    30.18964
...
```

Eigenvalues

- **Eigenvalue**

  An eigenvalue is a number, telling you how much variance there is in the data in that direction. The higher the number, the more variance a variable can potentially explain when contained in a model.

```
       eigenvalue variance.percent cumulative.variance.percent
Dim.1 2.71706780        30.1896422                    30.18964
Dim.2 2.62128516        29.1253906                    59.31503
Dim.3 2.42922025        26.9913361                    86.30637
Dim.4 0.36751103         4.0834559                    90.38982
Dim.5 0.33664969         3.7405521                    94.13038
Dim.6 0.27992699         3.1102999                    97.24068
Dim.7 0.08998043         0.9997825                    98.24046
Dim.8 0.08094112         0.8993458                    99.13981
Dim.9 0.07741752         0.8601947                   100.00000
```

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 4**

  Decide which new variables, i.e. PCA components, you want to keep for further analysis. But how?

  1. We want the **Eigenvalue** of a PCA component to be at least 1 as a value of 1 indicates that the variable explain at least as much variance in the data as it introduces itself
  2. Want to the **cumulative variance explained** of the retained PCA components to be at least 80%
  3. There should be **no huge breaks** of Eigenvalues between the PCA components we want to retain

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 4**

  Decide which new variables, i.e. PCA components, you want to keep for further analysis. But how?

cumu. variance explained >= 80%

Eigenvalue >= 1

	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	2.71706780	30.1896422	30.18964
Dim.2	2.62128516	29.1253906	59.31503
Dim.3	2.42922025	26.9913361	86.30637
Dim.4	0.36751103	4.0834559	90.38982

no huge breaks

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 5**

  We retain PCA components 1, 2, and 3 as new variables:

```
pcframe <- as.data.frame(pc$scores)[1:3]
```

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 6**

  Interpretation of the newly created variables by taking a closer look at their **loadings**:

```
> pc$loadings[,1:3]

Loadings:
      Comp.1 Comp.2 Comp.3
var1  0.316  0.168  0.414
var2 -0.376 -0.185 -0.435
var3 -0.353 -0.191 -0.417
var4        -0.474  0.260
var5         0.517 -0.301
var6         0.498 -0.285
var7 -0.433  0.222  0.273
var8  0.465 -0.251 -0.279
var9  0.458 -0.221 -0.272
```

The loadings of the PCA components are represented as **correlation values** of the individual PCA component with all variables that were part of the PCA

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 6**

  Interpretation of the newly created variables by taking a closer look at their **loadings**:

```
> pc$loadings[,1:3]

Loadings:
      Comp.1 Comp.2 Comp.3
var1   0.316  0.168  0.414
var2  -0.376 -0.185 -0.435
var3  -0.353 -0.191 -0.417
var4         -0.474  0.260
var5          0.517 -0.301
var6          0.498 -0.285
var7  -0.433  0.222  0.273
var8   0.465 -0.251 -0.279
var9   0.458 -0.221 -0.272
```

Apparently, our PCA components reflect the previous input variables quite nicely:

`Comp.1` includes the effects of `var7`, `var8`, and `var9`

`Comp.2` includes the effects of `var4`, `var5`, and `var6`

`Comp.3` includes the effects of `var1`, `var2`, and `var3`

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 6**

  Interpretation of the newly created variables by taking a closer look at their **loadings**:

```
> pc$loadings[,1:3]

Loadings:
     Comp.1 Comp.2 Comp.3
var1  0.316  0.168  0.414
var2 -0.376 -0.185 -0.435
var3 -0.353 -0.191 -0.417
var4        -0.474  0.260
var5         0.517 -0.301
var6         0.498 -0.285
var7 -0.433  0.222  0.273
var8  0.465 -0.251 -0.279
var9  0.458 -0.221 -0.272
```

Let's try a 'linguistic' interpretation:

Assume that
`var7` = speech rate
`var8` = phonological neighbourhood
`var9` = orthographical neighbourhood

`Comp.1` tells us that the effect of `var7` is opposite to the effect of `var8` & `var9`

Strategy 2: simulated data

- **Strategy 2**

  Use a principal component analysis to create new, non-correlated variables out of your correlated predictor variables. This is a more sophisticated, but also more labour-intensive way.

- **Step 6**

  Interpretation of the newly created variables by taking a closer look at their **loadings**:

```
> pc$loadings[,1:3]

Loadings:
     Comp.1 Comp.2 Comp.3
var1  0.316  0.168  0.414
var2 -0.376 -0.185 -0.435
var3 -0.353 -0.191 -0.417
var4        -0.474  0.260
var5         0.517 -0.301
var6         0.498 -0.285
var7 -0.433  0.222  0.273
var8  0.465 -0.251 -0.279
var9  0.458 -0.221 -0.272
```

Let's try a 'linguistic' interpretation:

Depending on the coefficient estimate of Comp.1 in a regression model, we thus find that

1. if it is positive: speech rate = lower value of dependent variable
2. if it is negative: speech rate = higher value of the dependent variable

Yes, PCAs are quite a handful…

Pre-Processing

- As the number of variables and the complexity of models increases, one has to consider a number of important factors

1. Variable distribution ✓
2. Assumptions of linear models ✓
3. Collinearity ✓