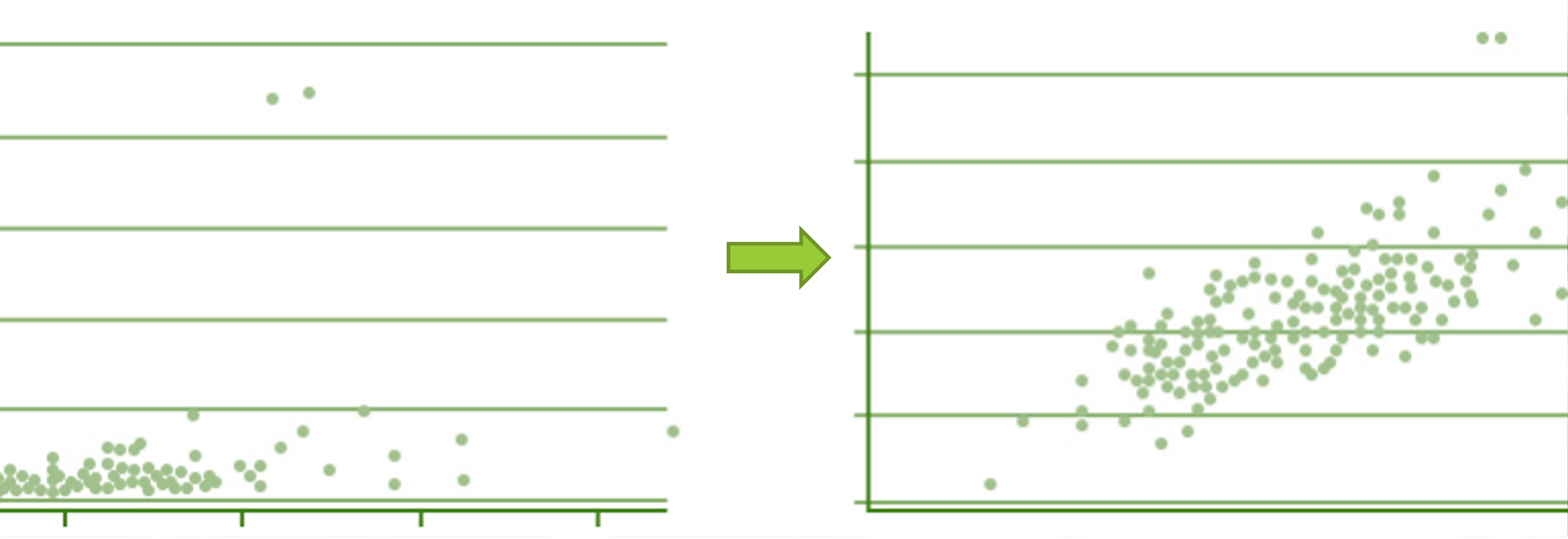


10. Dimensionality and Data Transformations

Dimensionality of Data

In data analysis, the **dimension** of the data is the number of attributes that are collected in a dataset, represented by the **number of columns**.

We can think of the number of variables used to describe each object (row) as a vector describing that object: the dimension is simply the **size** of that vector.

(**Note:** “dimension” is used differently in business intelligence contexts)

High Dimensionality and Big Data

Datasets can be "big" in a variety of ways:

- too large for the **hardware** to handle (cannot be stored, accessed, manipulated properly due to # of observations, # of features, the overall size)
- dimensions can go against **modeling assumptions** (# of features $\gg$ # observations)

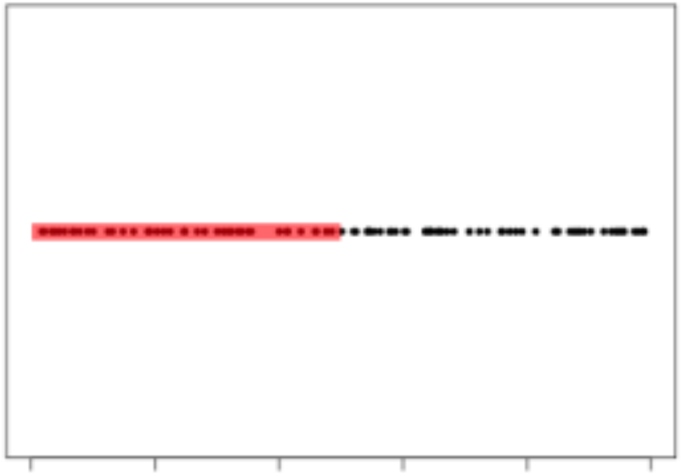
Examples:

- Multiple sensors recording 100+ observations per second in a large geographical area over a long time period = **very big dataset**
- In a corpus' *Term Document Matrix* (cols = terms, rows = documents), the number of terms is usually substantially higher than the number of documents, leading to **sparse data**

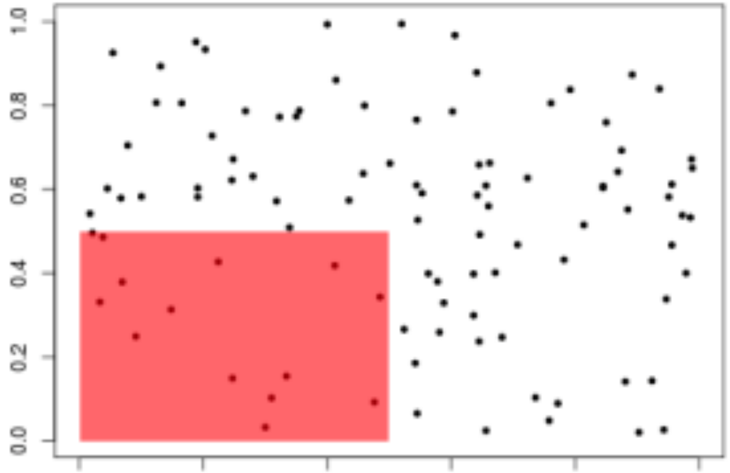
[http://simplystatistics.org]

Curse of Dimensionality

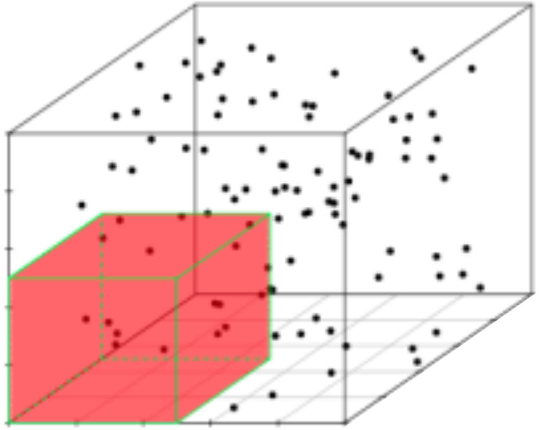
42% of data is captured



14% of data is captured



7% of data is captured



$N = 100$ observations, uniformly distributed on $[0,1]^d, d = 1, 2, 3.$
% of observations captured by $[0,1/2]^d, d = 1,2,3.$

Sampling Observations

Question: does every row of the dataset need to be used?

If rows are selected randomly (with or without replacement), the resulting sample might be **representative** of the entire dataset.

Drawbacks:

- if the signal of interest is rare, sampling might drown it altogether
- if aggregation is happening down the road, sampling will necessarily affect the numbers (passengers vs. flights)
- even simple operations on a large file (finding the # of lines, say) can be taxing on the memory – **prior information on the dataset structure can help**

Feature Selection

Removing **irrelevant**/**redundant** variables is a common data processing task.

Motivations:

- modeling tools do not handle these well (variance inflation due to multicolinearity, etc.)
- dimension reduction (# variables $\gg$ # observations)

Approaches:

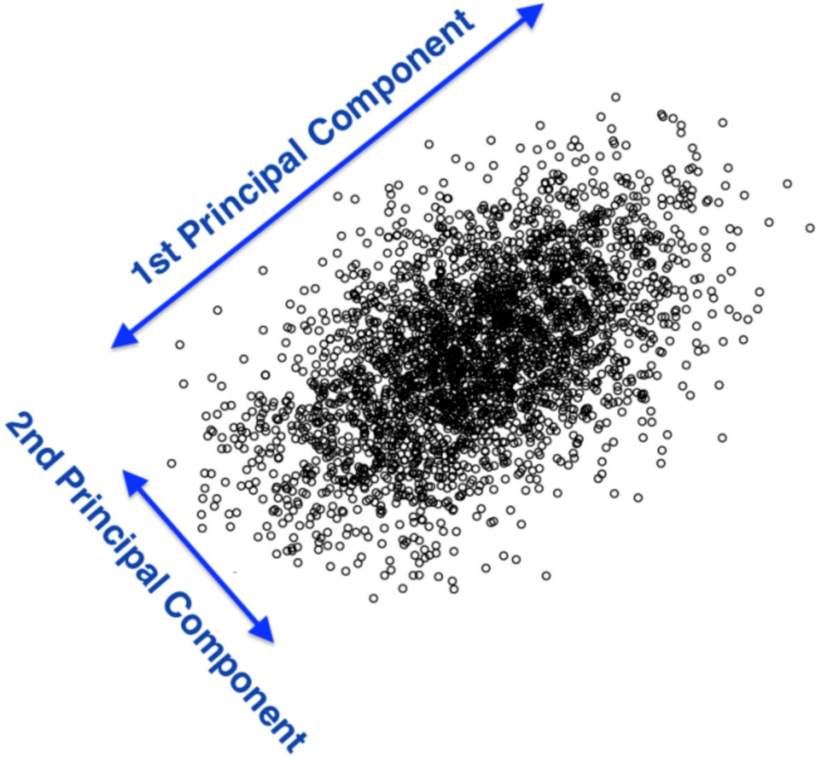
- filter vs. wrapper
- unsupervised vs. supervised

Dimension Reduction: PCA

Motivational Example: Nutritional Content of Food

What is the best way to differentiate food items? Vitamin content, fat, or protein level? A bit of each?

Principal Component Analysis (PCA) can be used to find the combinations of variables along which the data points are **most spread out** (dimension reduction).



[A. Ng, K. Soo, *Numsense!*]

Dimension Reduction: PCA

Presence of nutrients appears to be **correlated** among food items.

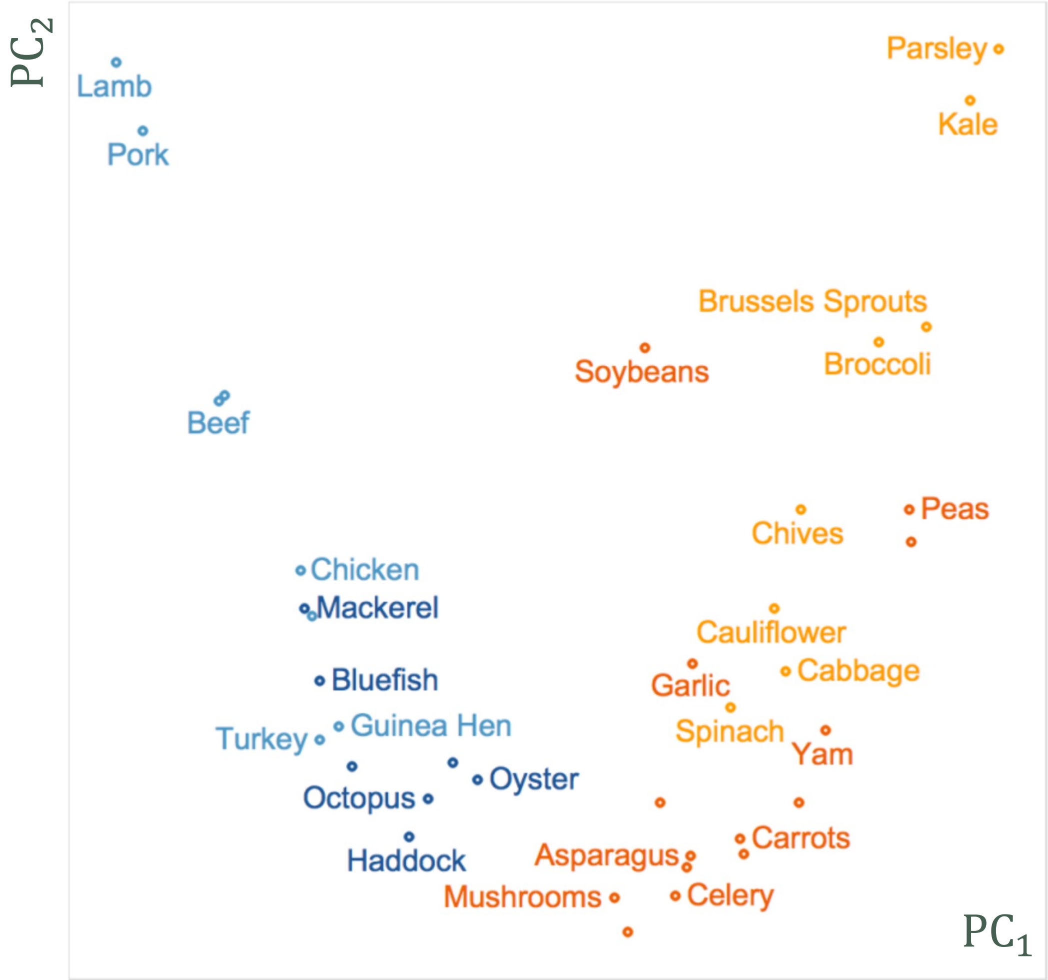
In the (small) sample consisting of Lamb, Pork, Kale, and Parsley, *Fat* and *Protein* levels seem in step, as do *Fiber* and *Vitamin C*.

In a larger dataset, the correlations are $r = 0.56$ and $r = 0.57$.

How much could 2 **derived** variables explain?

$$PC_1 = -\mathbf{0.45} \times \mathbf{Fat} - \mathbf{0.55} \times \mathbf{Protein} + \mathbf{0.55} \times \mathbf{Fiber} + \mathbf{0.44} \times \mathbf{Vitamin\ C}$$
$$PC_2 = \mathbf{0.66} \times \mathbf{Fat} + 0.21 \times \text{Protein} + 0.19 \times \text{Fiber} + \mathbf{0.70} \times \mathbf{Vitamin\ C}$$

PCA Differentiation



PC_1 differentiates vegetables from meats; PC_2 differentiates 2 **sub-categories** within these:

- **meats** are concentrated on the left (low PC_1 values)
- **vegetables** are concentrated on the right (high PC_1 values)
- **seafood** have lower *Fat* content (low PC_2 values) and are concentrated at the bottom
- **non-leafy veggies** have lower *Vitamin C* content (low PC_2 values) and are also bunched at the bottom

Common Transformations

Models sometimes require that certain data assumptions be met (normality of residuals, linearity, etc.).

If the raw data does not meet the requirements, we can either:

- abandon the model
- attempt to **transform** the data

The second approach requires an **inverse transformation** to be able to draw conclusions about the **original data**.

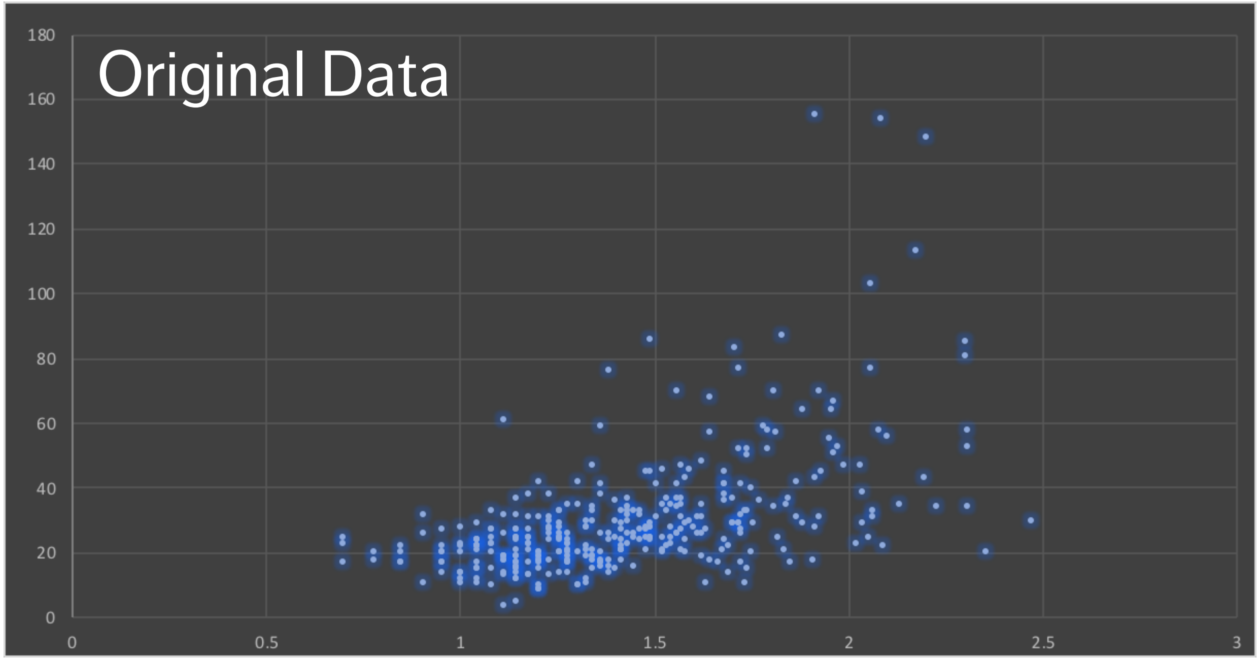
Common Transformations

In the data analysis context, transformations are **monotonic:**

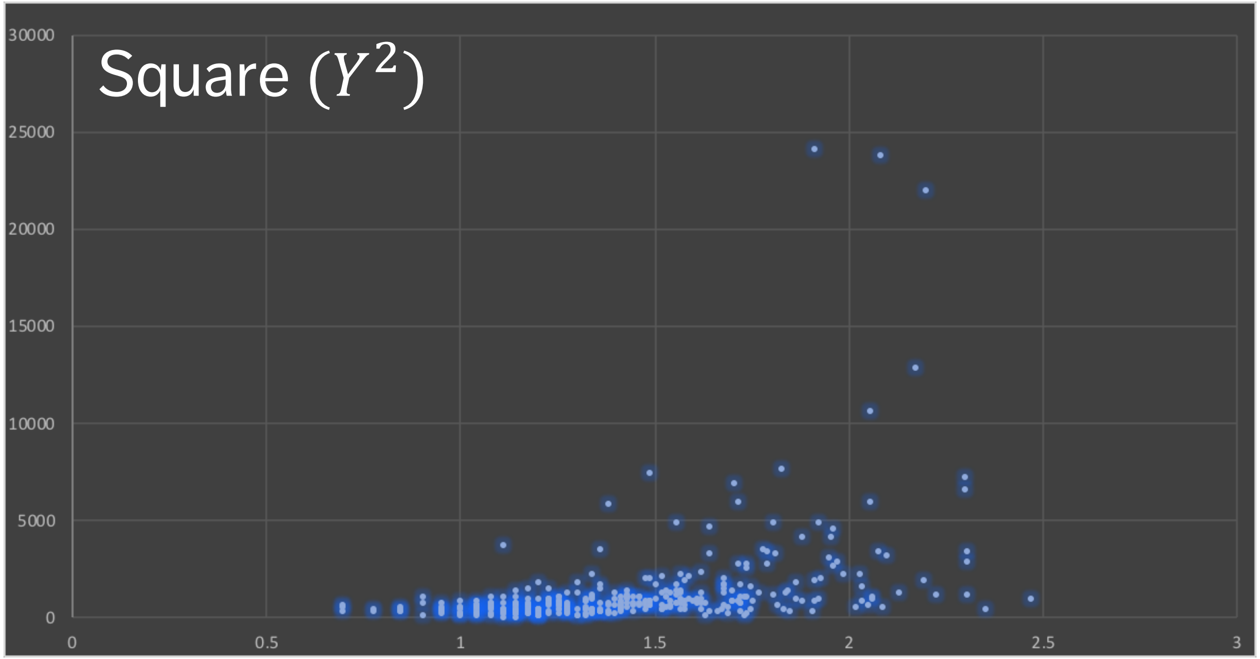
- logarithmic
- square root, inverse, power: W^k
- exponential
- Box-Cox, etc.

Transformations on X may achieve linearity, but usually at some price (correlations are not preserved, for instance). Transformations on Y can help with non-normality and unequal variance of error terms.

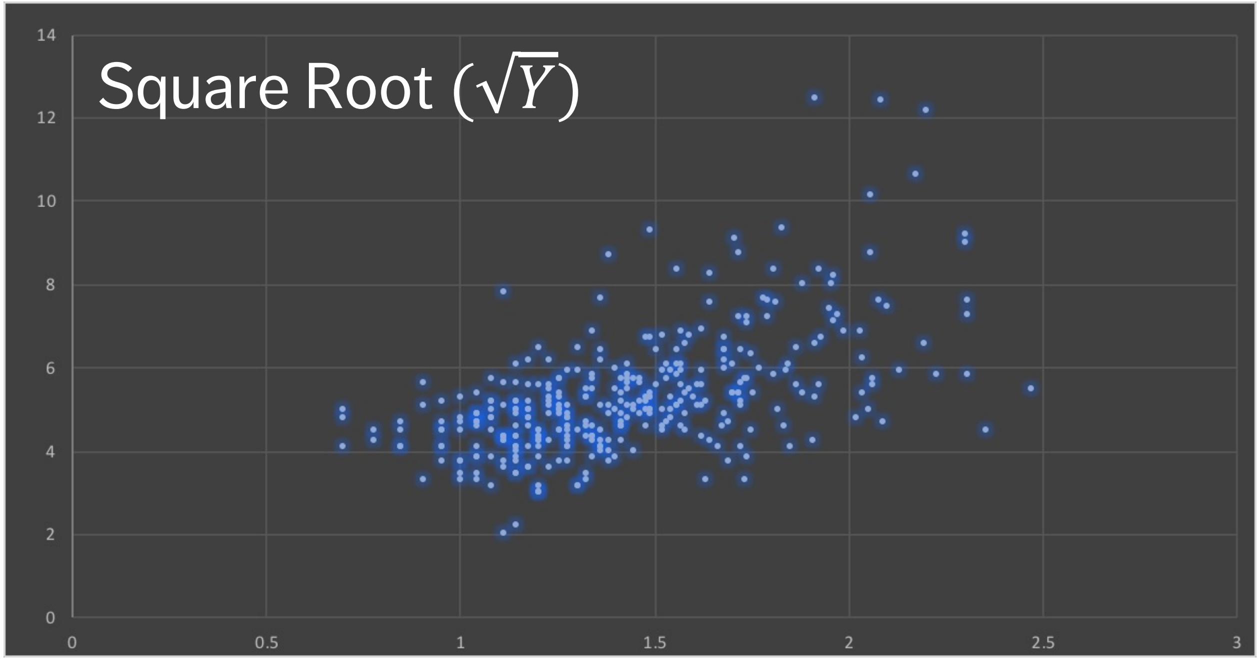
Session 4



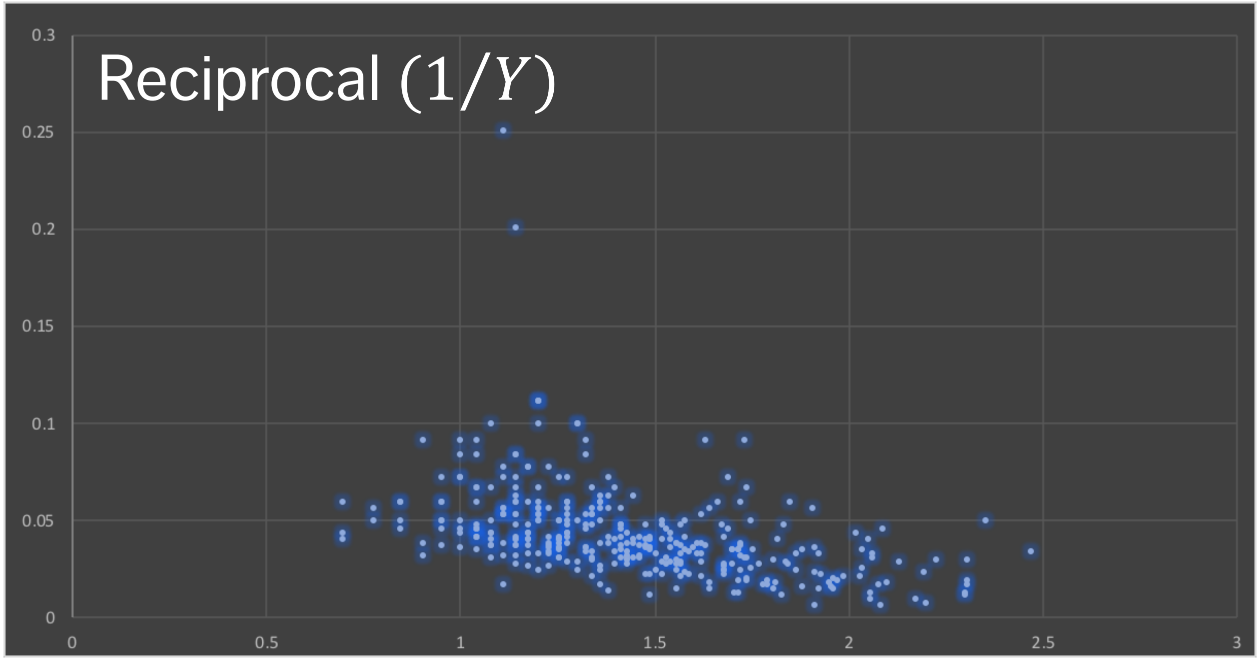
Original Data
180
160
140
120
100
80
60
40
20
0
0
0.5
1
1.5
2
2.5
3



Square (Y^2)
30000
25000
20000
15000
10000
5000
0
0
0.5
1
1.5
2
2.5
3



Square Root ($\sqrt{Y}$)
14
12
10
8
6
4
2
0
0
0.5
1
1.5
2
2.5
3



Reciprocal ($1/Y$)
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.5
1
1.5
2
2.5
3

gm = geometric mean

Box-Cox Transformation

Assume the usual model $Y_j = \sum_i \beta_i X_{j,i} + \varepsilon_j$ with either

- skewed residuals
- not-constant variance
- non-linear trend

The **Box-Cox transformation** $Y_j \mapsto Y_j'(\lambda)$ suggests a choice: select λ which maximizes the corresponding log-likelihood

$$Y_j'(\lambda) = \begin{cases} \mathrm{gm}(\boldsymbol{Y}) \times \ln(Y_j), & \lambda = 0 \\ \lambda^{-1} \mathrm{gm}(\boldsymbol{Y})^{1-\lambda} \times (Y_j^{\lambda} - 1), & \lambda \neq 0 \end{cases}$$

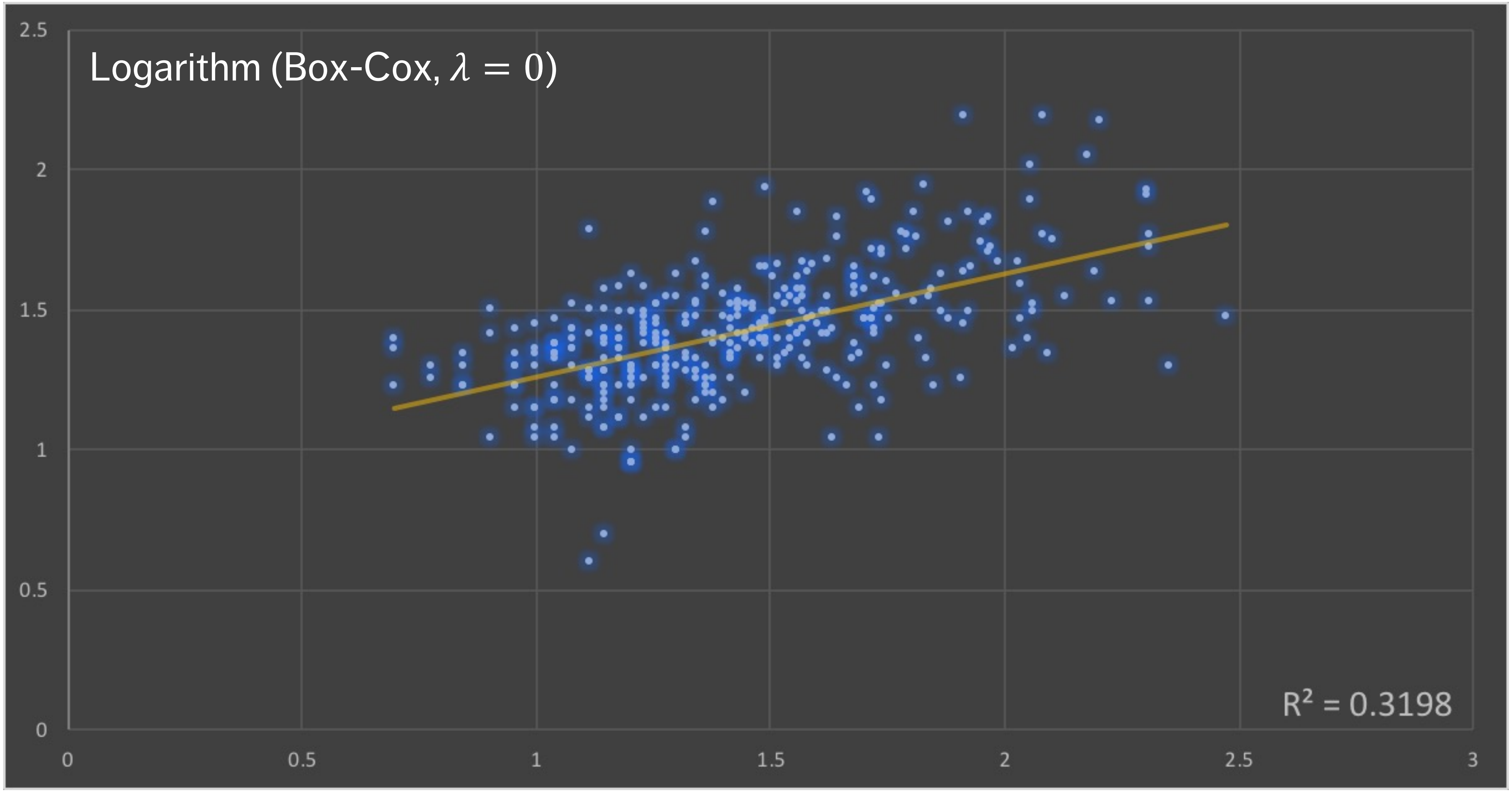
Box-Cox Transformation

The procedure provides a **guide** to select a transformation.

Theoretical/practical **rationales** may exist for a particular choice of λ.

Residual analysis is still required to ensure that the choice was appropriate.

Better to work with (interpret) the transformed data.


Logarithm (Box-Cox, $\lambda = 0$)
$R^2 = 0.3198$
2.5
2
1.5
1
0.5
0
0
0.5
1
1.5
2
2.5
3

Scaling

Numeric variables may have different **scales** (i.e., weights and heights).

The variance of a large-range variable is typically greater than that of a small-range variable, introducing a bias (for instance).

Standardization creates a variable with mean 0 and std. dev. 1:

$$Y_i = \frac{X_i - \bar{X}}{s_X}$$

Normalization creates a new variable in the range $[0,1]$: $Y_i = \frac{X_i - \min X}{\max X - \min X}$

Discretizing

To reduce computational complexity, a numeric variable may need to be replaced by an **ordinal** variable (from *height* value to "*short*", "*average*", "*tall*", for instance).

Domain expertise can be used to determine the bins' limits (although that may introduce unconscious bias to the analyses)

In the absence of such expertise, limits can be set so that either

- the bins each contain the same number of observations
- the bins each have the same width
- the performance of some modeling tool is maximized

Creating Variables

New variables may need to be introduced:

- as **functional relationships** of some subset of available features
- because modeling tool may require **independence of observations**
- because modeling tool may require **independence of features**
- to simplify the analysis by looking at **aggregated summaries** (often used in text analysis)

Time dependencies $\longrightarrow$ time series analysis (lags?)

Spatial dependencies $\longrightarrow$ spatial analysis (neighbours?)

Suggested Reading

Dimensionality and
Data Transformations

Data Understanding, Data Analysis, Data Science
Data Preparation

Data Transformations

- Common Transformations
- Box-Cox Transformations
- Scaling
- Discretizing
- Creating Variables

*__Feature Selection and Dimension Reduction__ (advanced)

Exercises

Dimensionality and
Data Transformations

1. Using Example: Algae Bloom as a basis, scale, discretize, and create new variables out of the algae blooms dataset.
2. Scale, discretize, and create new variables out of the grades and cities.txt datasets.
3. Scale, discretize, and create new variables out of a dataset of your choice.