FIT5196 DATA WRANGLING

Week 9

Data Transformation

By Kiara Wang & Jackie Rong

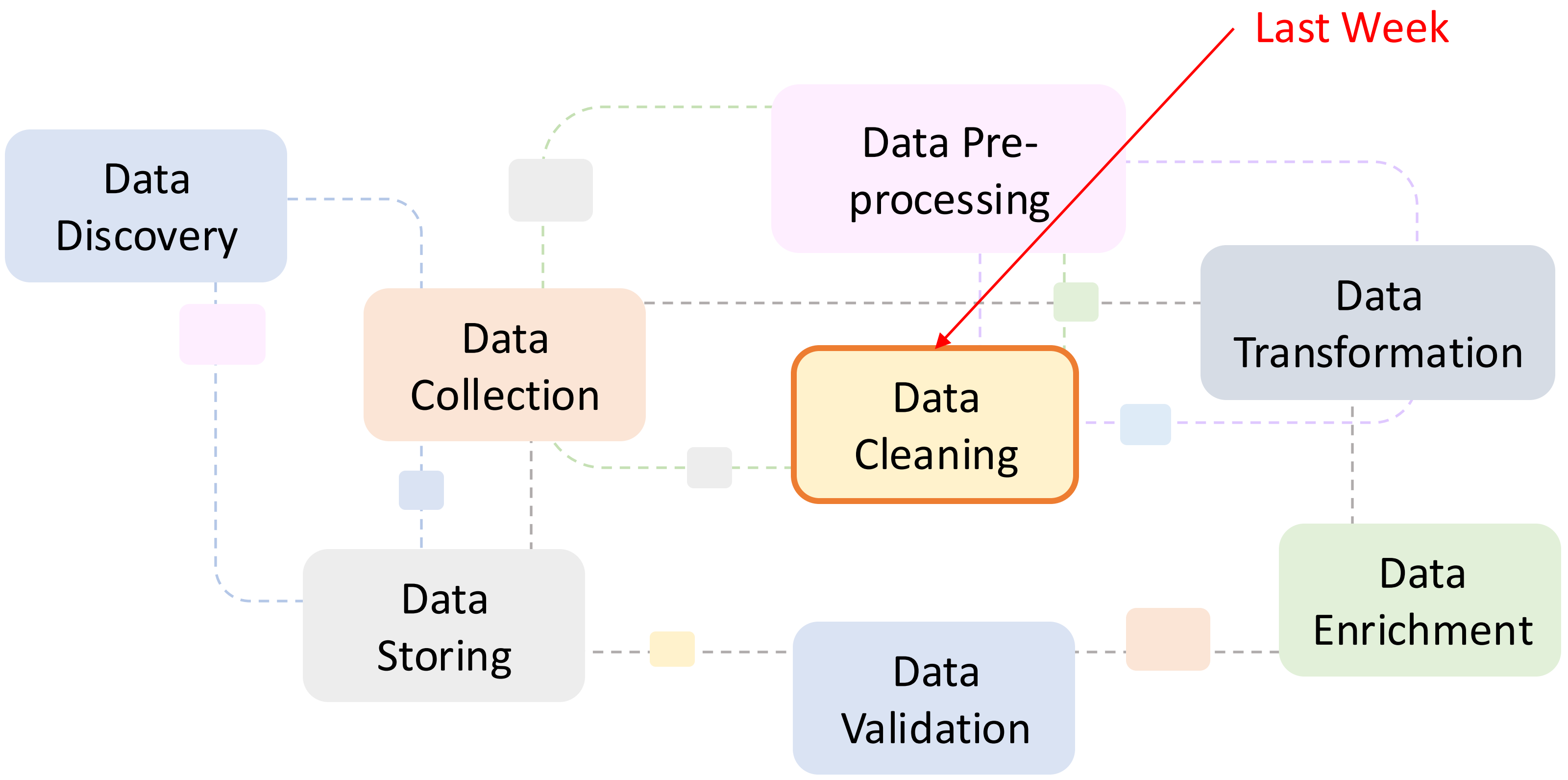
Faculty of Information Technology

Monash University

Data Wrangling Tasks (Recap)

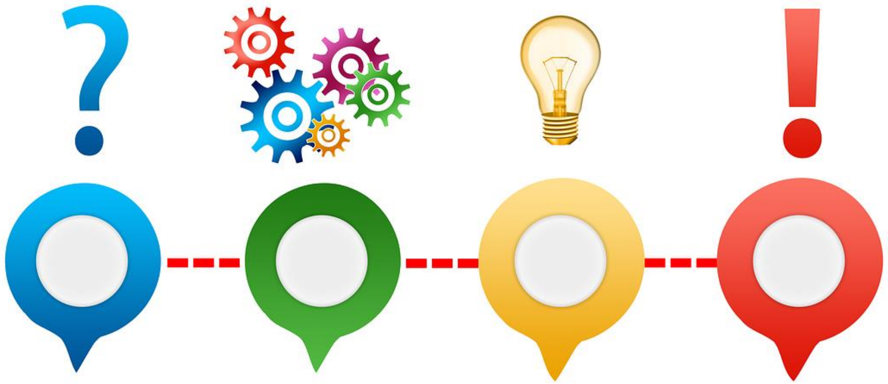
In the **Data Pre-processing** stage, preliminary data preparation tasks are performed to make raw data more suitable for analysis.

- Deal with missing data
- Deal with outliers



Data Transformation

- Overview of Data Transformation
- Data Normalisation
- Data Discretisation
- Data Construction
  - Feature Engineering
  - Data Sampling



Data Transformation

- **Data transformation** involves cleaning and converting raw data into a format that is more suitable for analysis.
- The **goal** of data transformation is to ensure the data is in usable and efficient format that makes analysis straightforward and reliable.
- Reasons for data transformation
  - Fix skewness in data
  - Enhance data visualisation
  - Better interpretability
  - Improve the compatibility of data with assumptions underlying a modelling process

Data Transformation

- Data transformation involves
  - Data Normalisation
  - Linear Transformation
  - Power Transformation
  - Data Discretisation
  - Data Construction
  - Data Reduction

Data Normalisation

- **Data normalization** is a pivotal aspect of data preparation, particularly important when preparing data for machine learning and statistical analysis.
- The **purpose** of normalisation is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values or losing information.
- Normalization is crucial when features have different units (like dollars, kilometres, and hours) or vary widely in scale.
- There are two types of data normalisation
  - Scaling
  - Standardisation

Scaling

- **Scaling** focuses on rescaling data value range to a specific interval.
  - Min-Max scaling
  - MaxAbs scaling
  - Decimal scaling
  - Robust scaling
  - Log scaling

Min-Max Scaling

- **Min-Max Scaling** (also known as **normalization**) is one of the simplest methods and involves rescaling the range of features to scale the range in [0, 1] or [-1, 1].

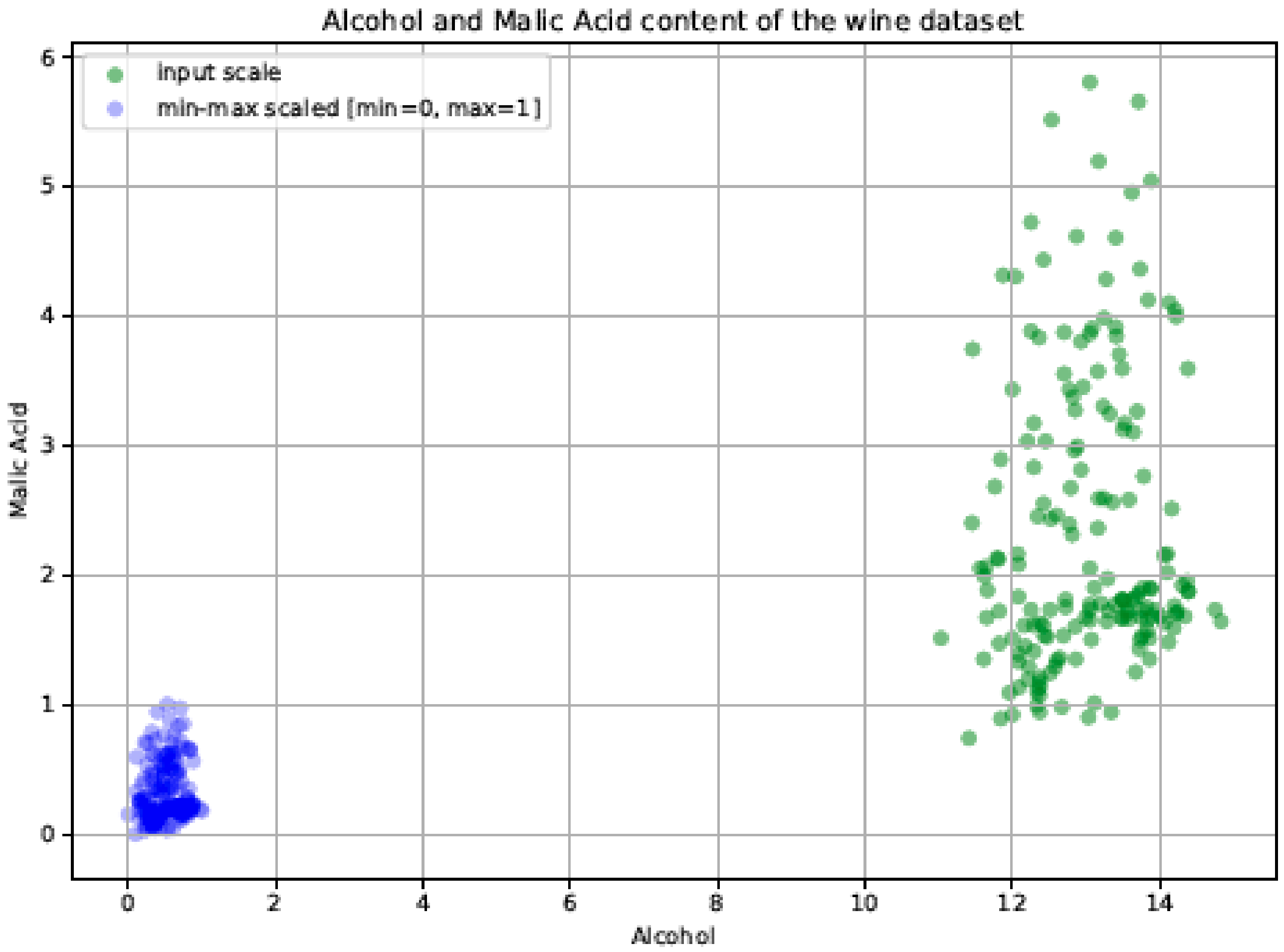
$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

where x_{min} is the minimum value and x_{max} is the maximum value in the column.

- This method can be expanded to be a general scaling in [n, m]

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}(m - n) + n$$

Min-Max Scaling



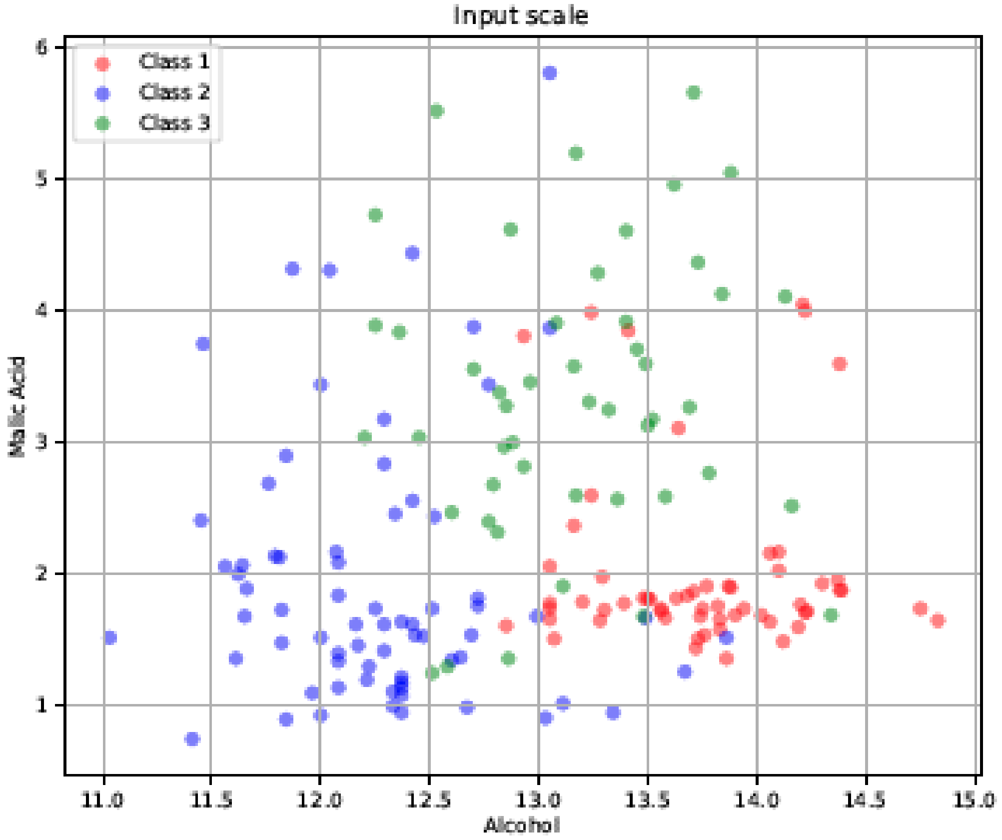
Pros:

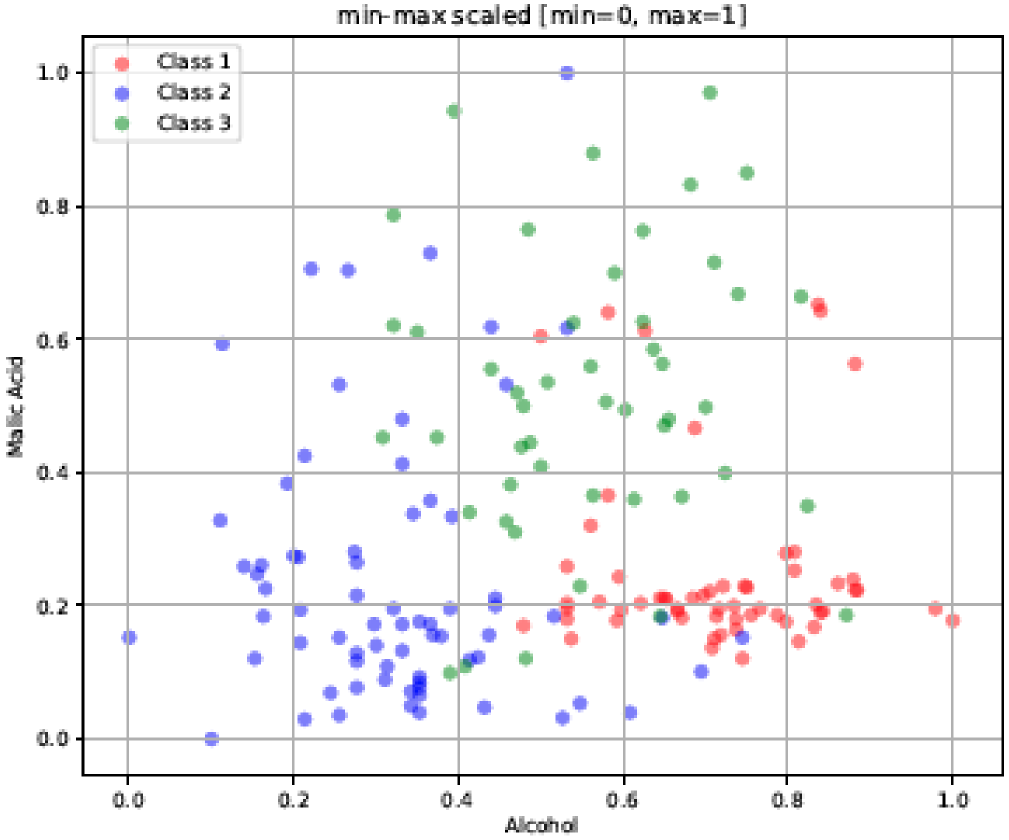
- Easy to implement and understand.
- Preserves the original distribution of scores, except for a scaling factor.

Cons:

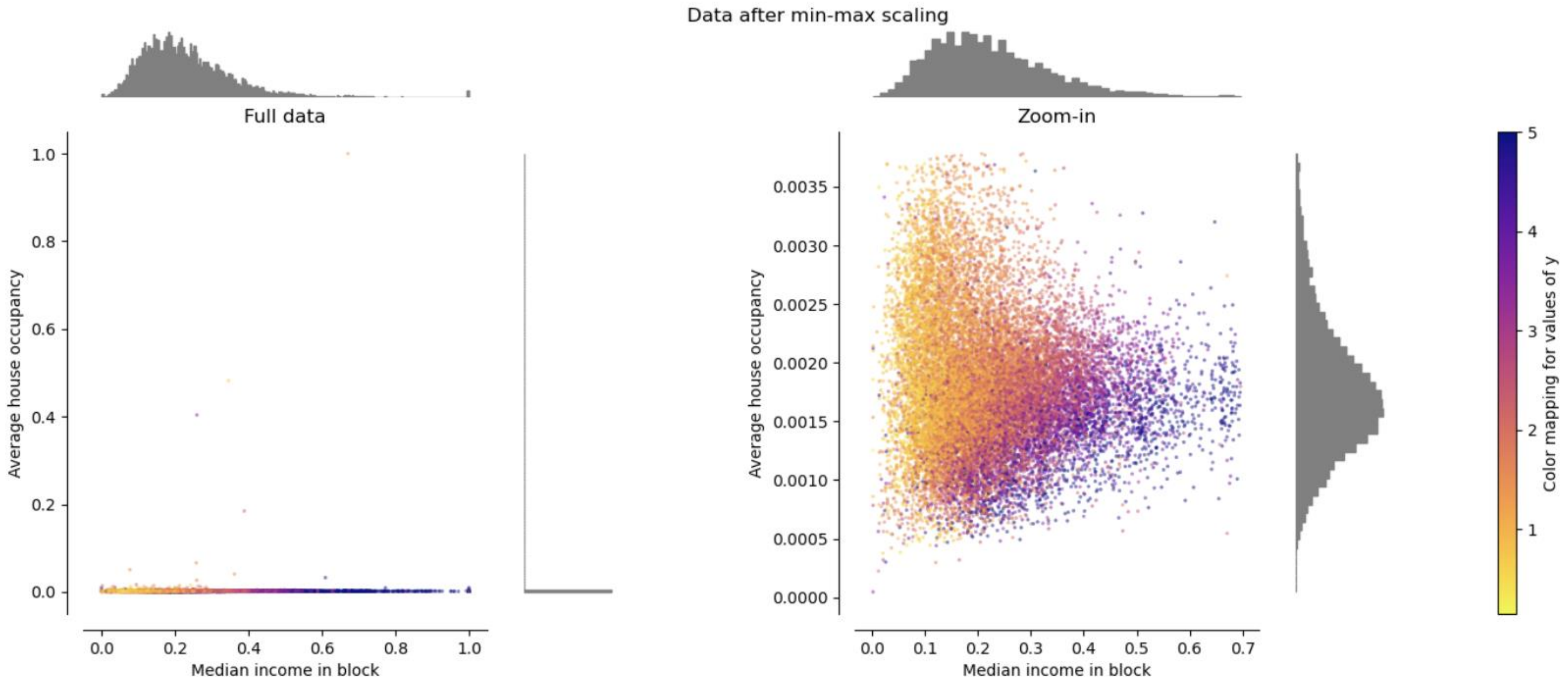
- Sensitive to outliers. If an outlier is present, then all other values are squeezed in a narrow range.

Min-Max Scaling





Min-Max Scaling



MaxAbs Scaling

- MaxAbs Scaling scales each feature by its maximum absolute value to be in the range [-1, 1].
- This is done by dividing each value by the maximum absolute value in the feature.

$$x_{scaled} = \frac{x}{\max(|x|)}$$

- Example

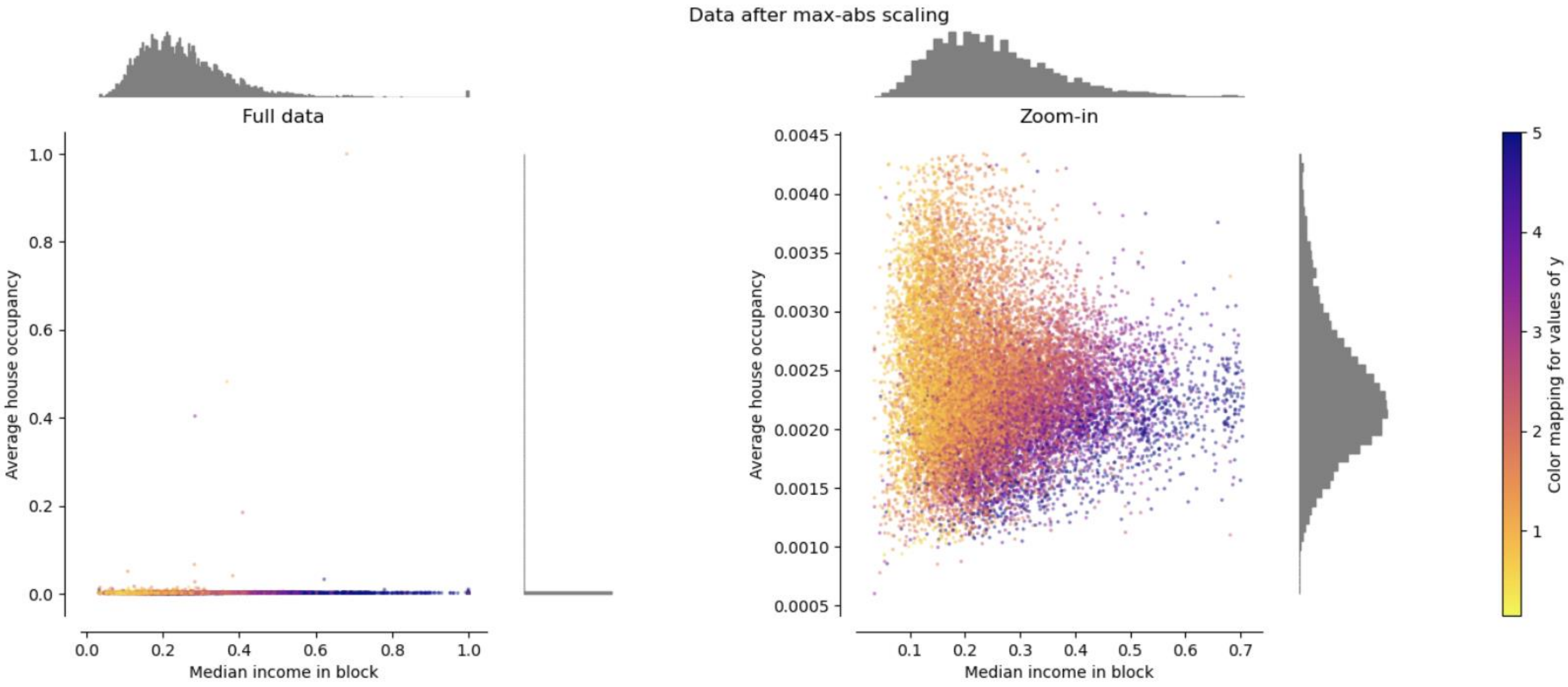
Given $x = [-10, 5, 20, -15]$, apply MaxAbs scaling

- $x_{max} = \max(abs(x)) = 20$
- $x_{scaled} = \frac{x}{20}$

Pros
- This method does not shift/centre the data, and thus does not destroy any sparsity.
- Simple and quick to apply.

Cons
- Sensitive to Outliers.

MaxAbs Scaling



Decimal Scaling

- Shift the decimal place of a numeric value such that the maximum absolute value will always be less than 1.

$$x_{scaled} = \frac{x}{10^c}$$

where c is the smallest integer such that $\max(|x_{scaled}|) < 1$.

- Example

$$-500 \leq x \leq 45 \Longrightarrow -0.500 \leq x \leq 0.045$$

  - $x_{max} = \max(abs(x)) = 500$
  - $c = ceil(\log_{10} x_{max}) = 3.0$
  - $x_{scale} = \frac{x}{10.0^3} = \frac{x}{1000}$

Pros
- Simple and Intuitive.
- Preserves Original Relationships.

Cons
- Sensitive to Outliers.
- Limited Standardization Impact.

Robust Scaling

- Robust Scaling uses the median and the quartile range (defined as the difference between the 75th and 25th quartiles).

$$x_{scaled} = \frac{x - x_{median}}{IQR(x)}$$

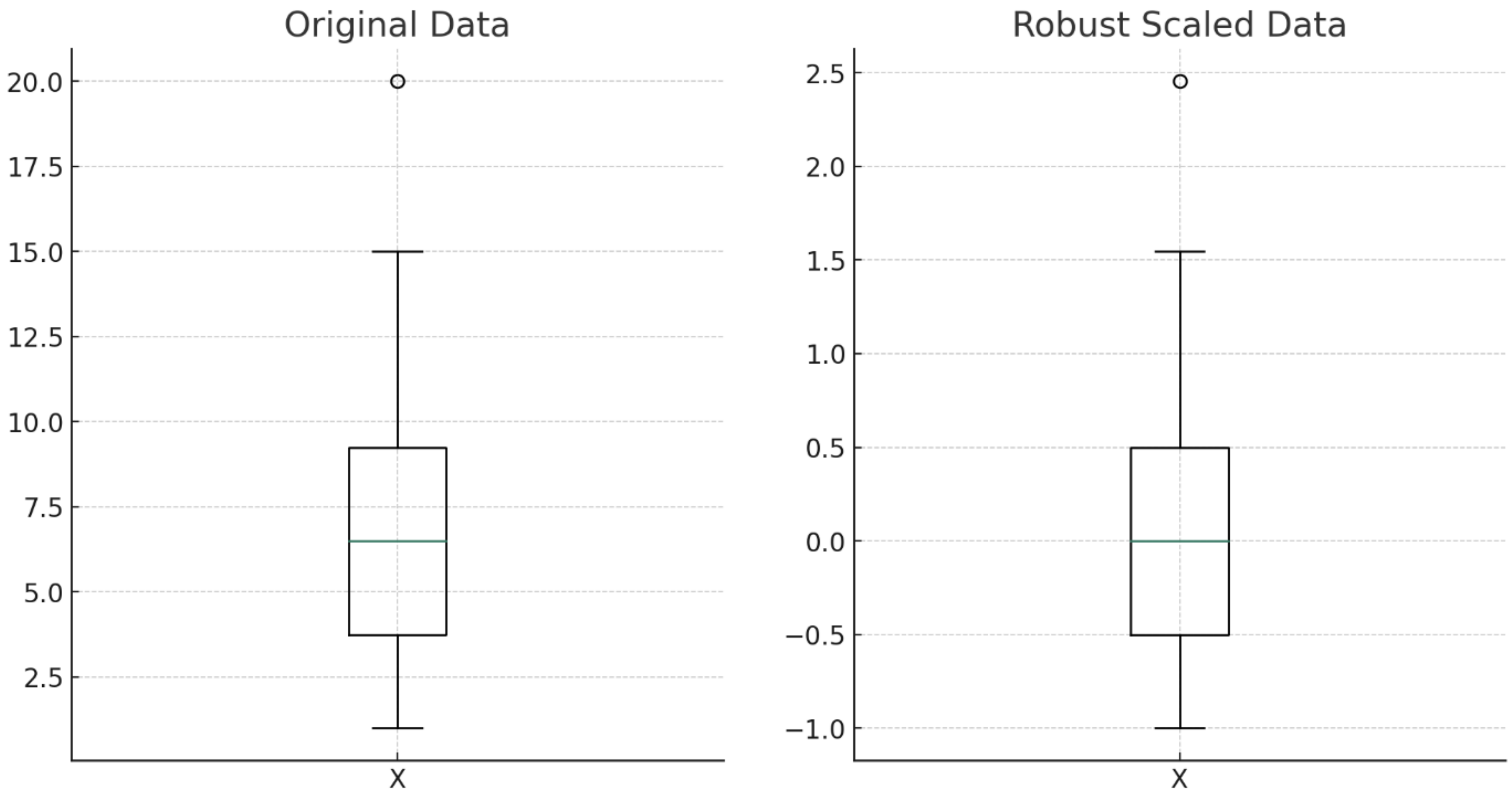
$$IQR(x) = Q3(x) - Q1(x)$$

where $Q1$ and $Q3$ are the 25th and 75th quartiles, respectively.

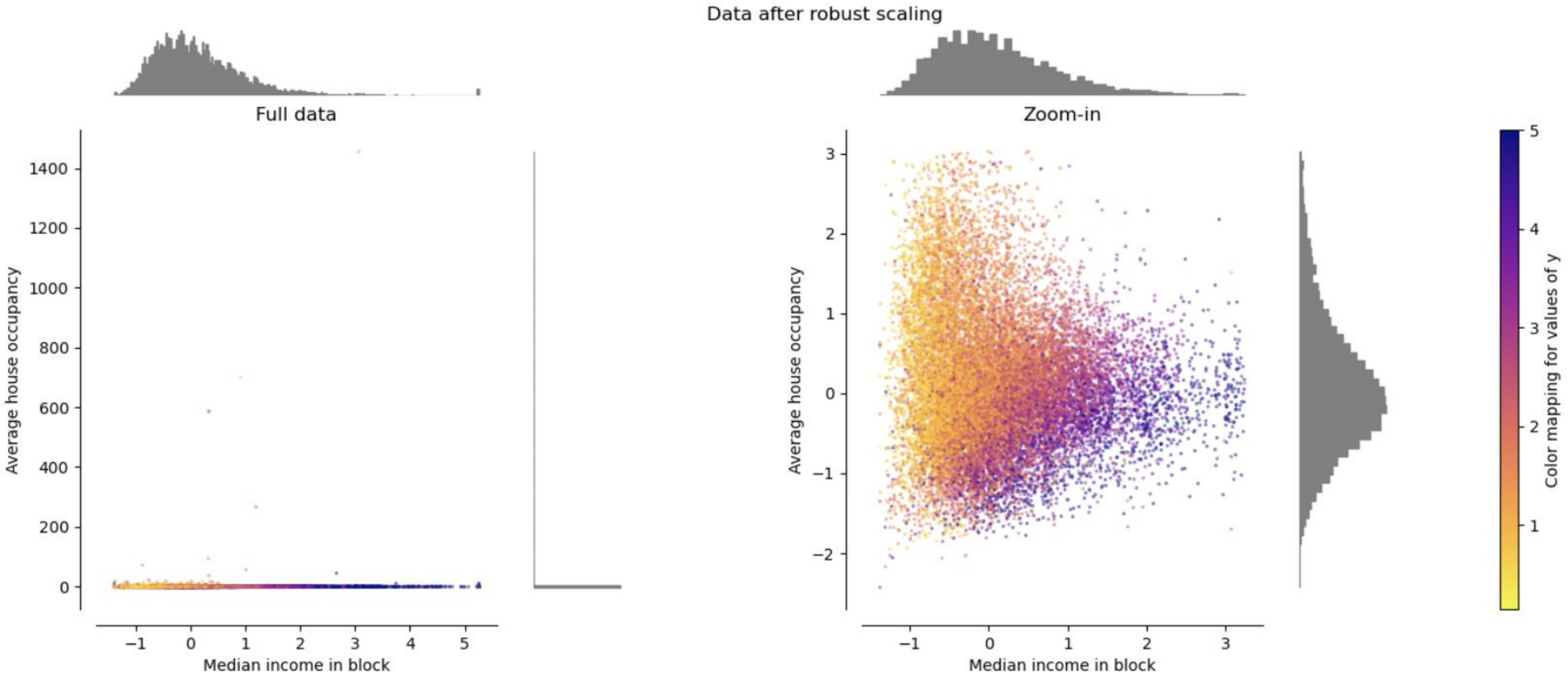
Pros:
- Very robust to outliers.
- Effective scale normalization maintains a more useful data distribution and scale.

Cons:
- Quartile computation can be computationally more expensive than mean/standard deviation.

Robust Scaling



Robust Scaling



Log Scaling

- Logarithmic scaling can be useful when the data involves exponential growth (e.g., population growth, viral spread).
- The log transform can help stabilize the variance and make the data more "normal".

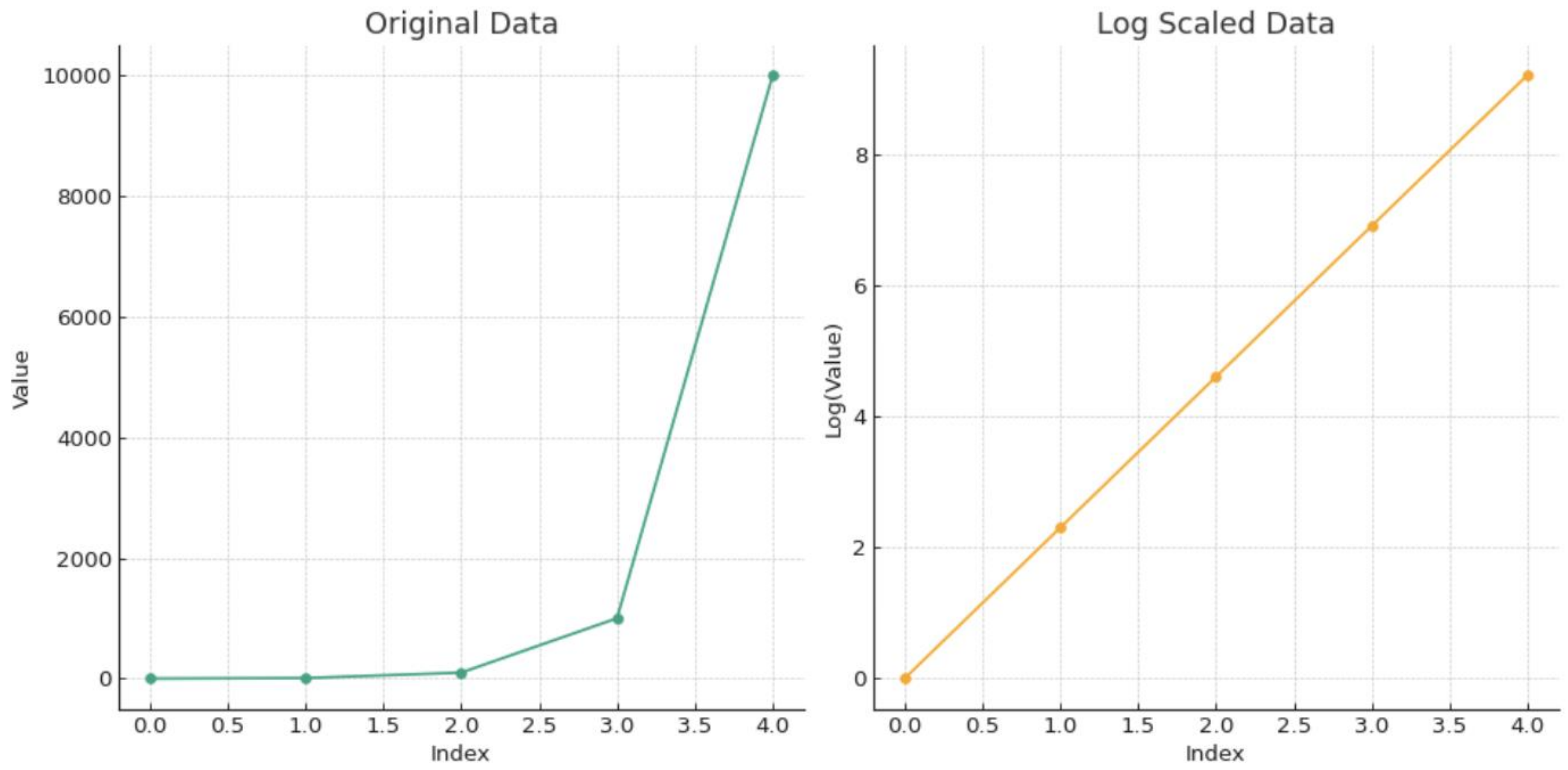
$$x_{scaled} = \log(x)$$

where $\log()$ is the natural logarithm of another base, depending on the data distribution.

Pros:
- Reduces Skewness.
- Stabilizes Variance.

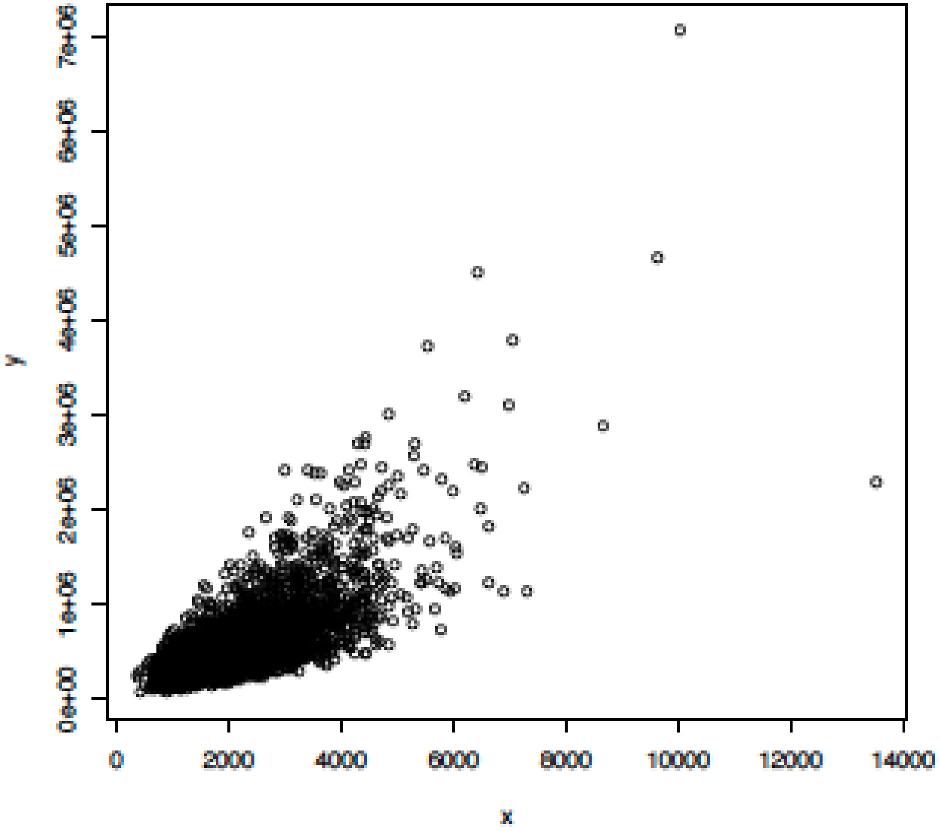
Cons:
- Limited to Positive Values.
- Can Obscure Small Differences.

Log Scaling

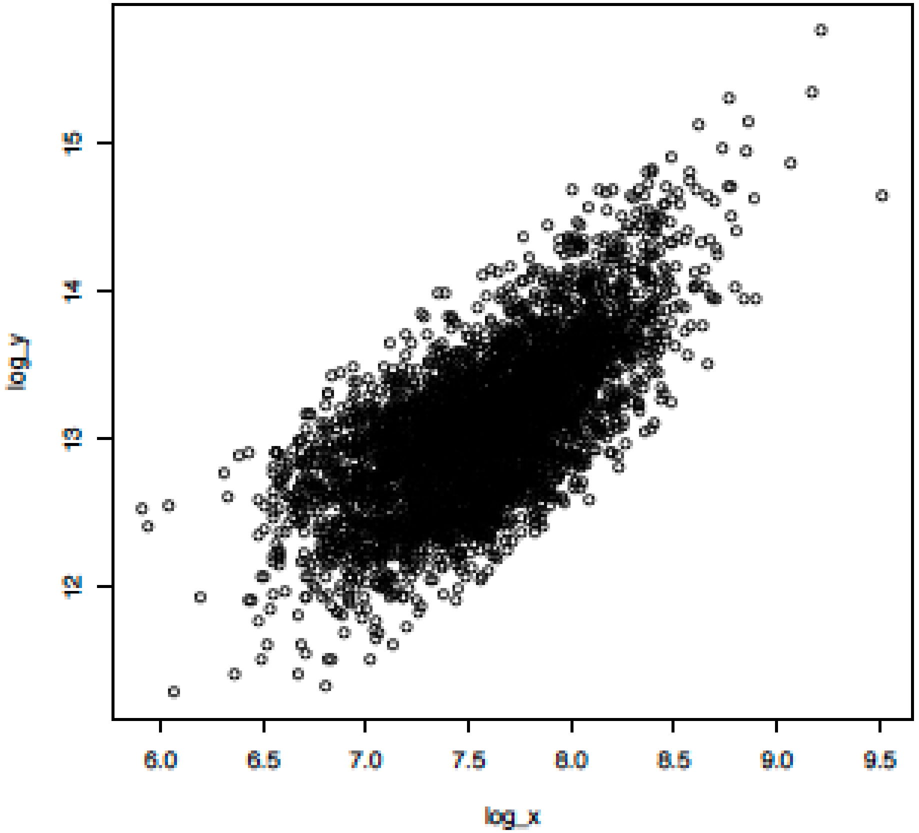


Original Data
Value
10000
8000
6000
4000
2000
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Index
Log Scaled Data
Log(Value)
8
6
4
2
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Index

Log Scaling



y
7e+06
6e+06
5e+06
4e+06
3e+06
2e+06
1e+06
0e+00
0
2000
4000
6000
8000
10000
12000
14000
x



log_y
15
14
13
12
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
log_x

Scaling

- **Scaling** focuses on rescaling data value range to a specific interval.
  - Min-Max scaling
  - MaxAbs scaling
  - Decimal scaling
  - Robust scaling
  - Log scaling

Standardisation

- **Standardisation** (**z-score normalisation**) involves rescaling the data to have a mean (average) of 0 and a standard deviation of 1.

$$\mu = 0, \sigma = 1.0$$

- The new value can be calculated by

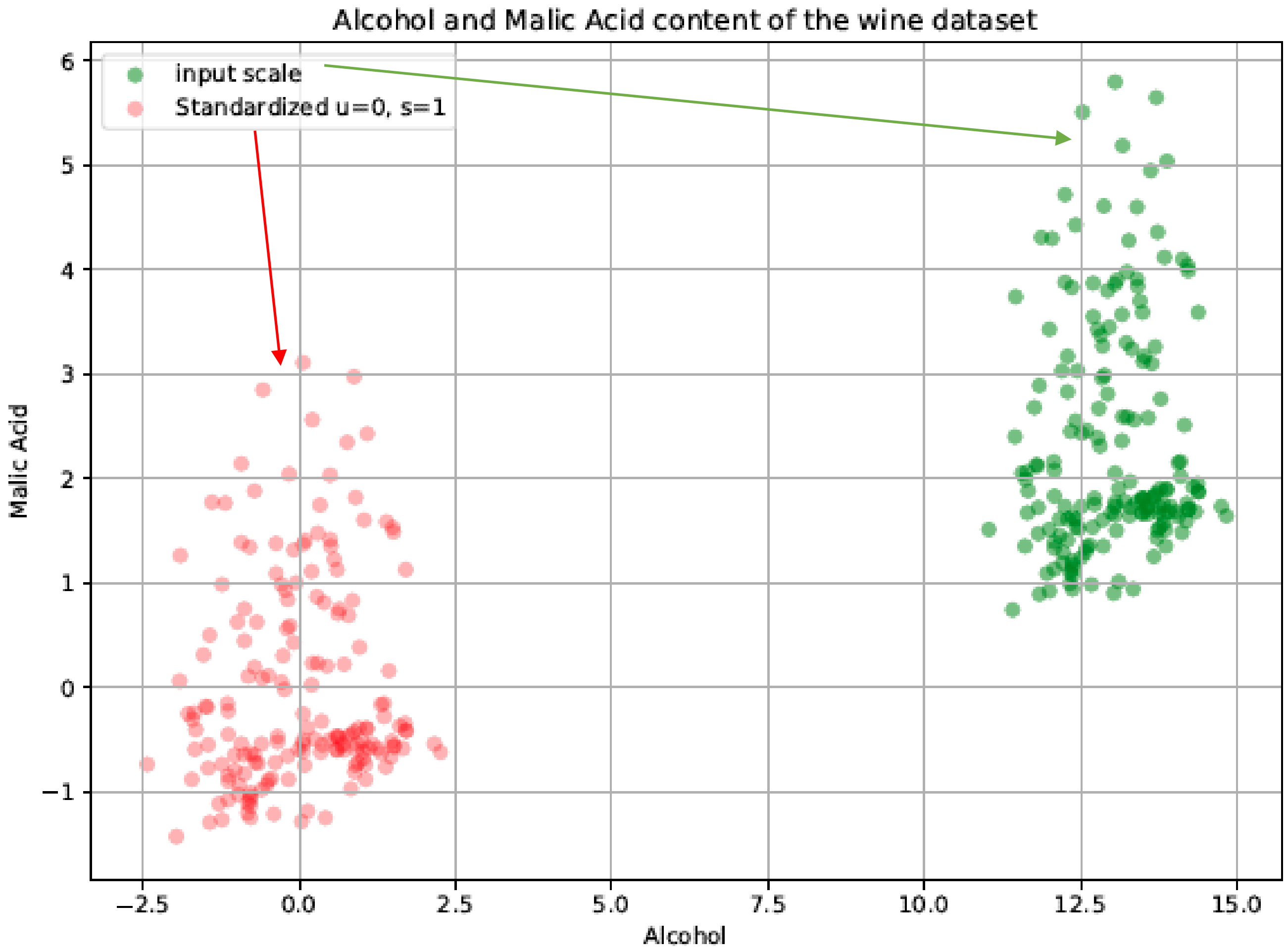
$$z = \frac{x - \mu}{\sigma}$$

where

$$\mu = \frac{1}{n}\sum_i x_i$$

$$\sigma = \sqrt{\frac{1}{n}\sum_i (x_i - \mu)^2}$$

Standardisation



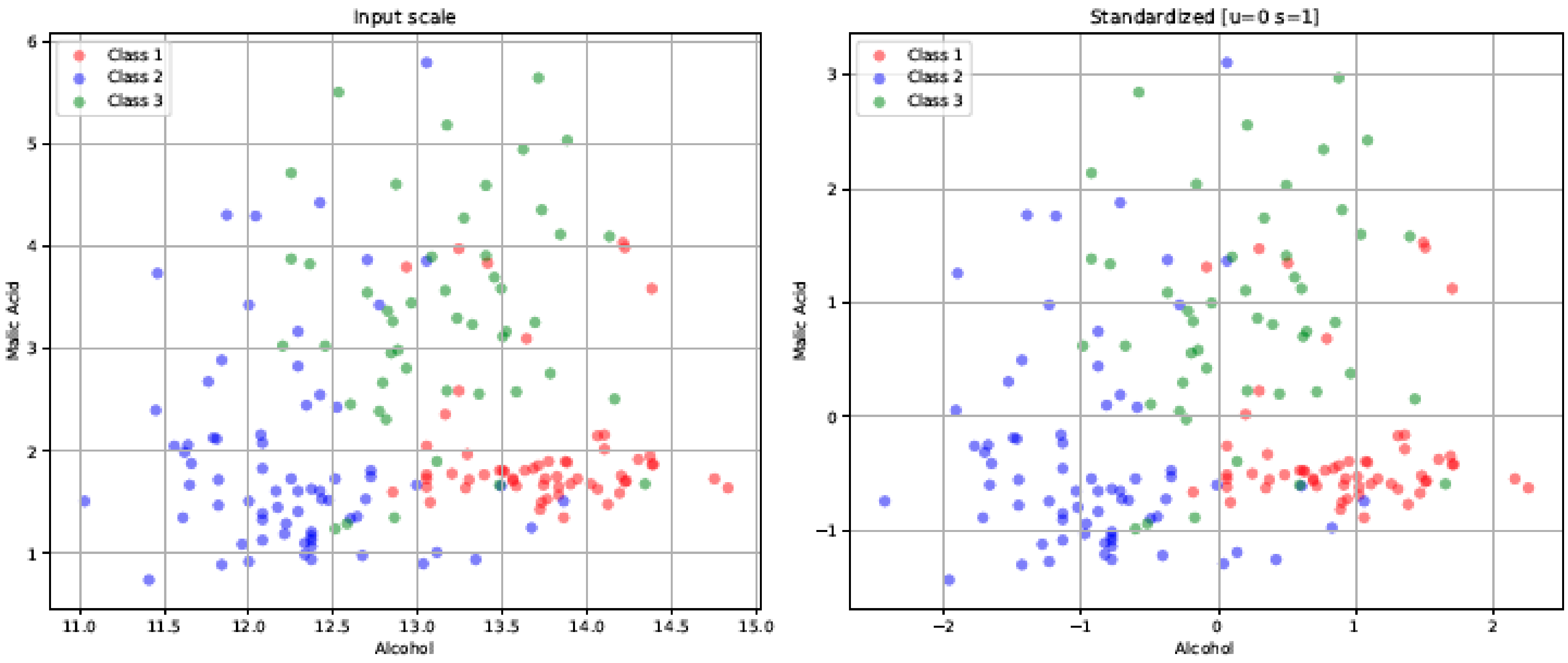
Pros:

- Handles outliers better than Min-Max scaling.
- Useful in algorithms that assume data is normally distributed.

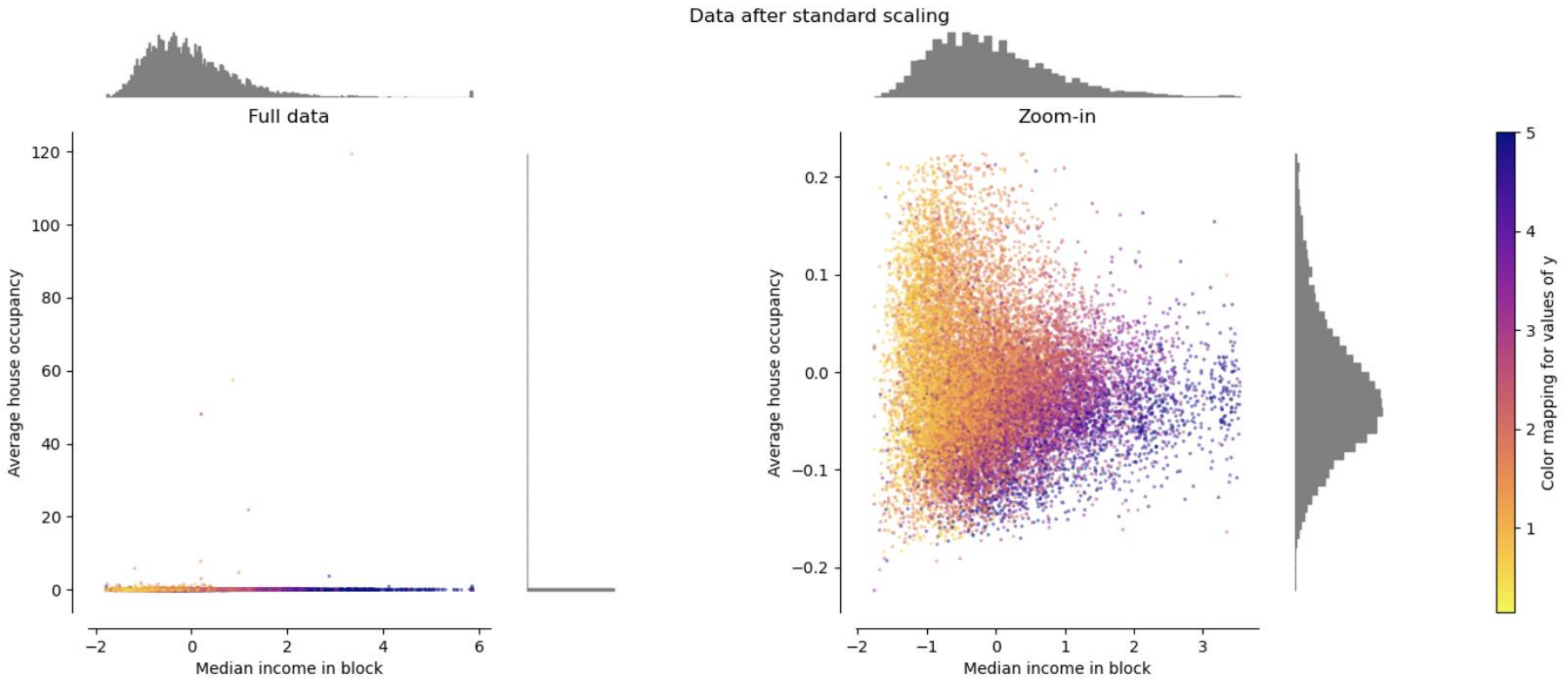
Cons:

- Does not produce normalized data with the exact same scale as Min-Max scaling.

Standardisation



Standardisation



Linear Transformation

- Linear transformation preserves the linear relationship between the features.
- Aggregate the information contained in various features.
- Given a subset of the complete set of attributes $x_1, x_2, \ldots, x_m$,

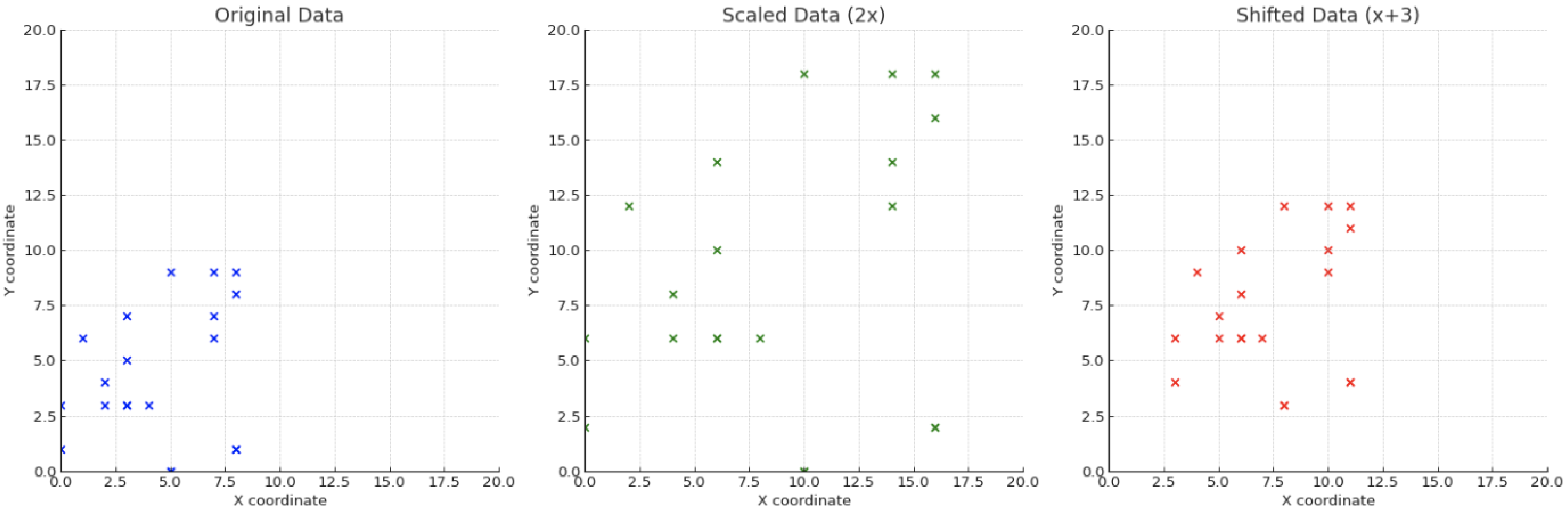
$$x_{linear} = w_0 + \sum_{i=1}^{m} w_i x_i$$

- Example
  - Celsius to Fahrenheit
  - Miles to Kilometres
  - Inches to Centimetres

Pros:
- Simple and clear.
- Enhances comparability.

Cons:
- Outlier sensitivity.
- Non-linearity and distribution limits.

Linear Transformation

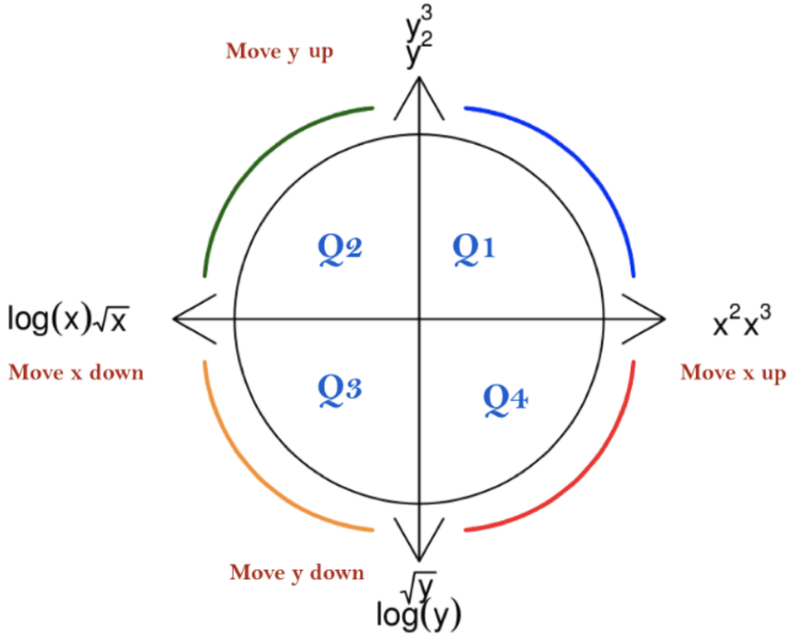


MONASH University

Power Transformation

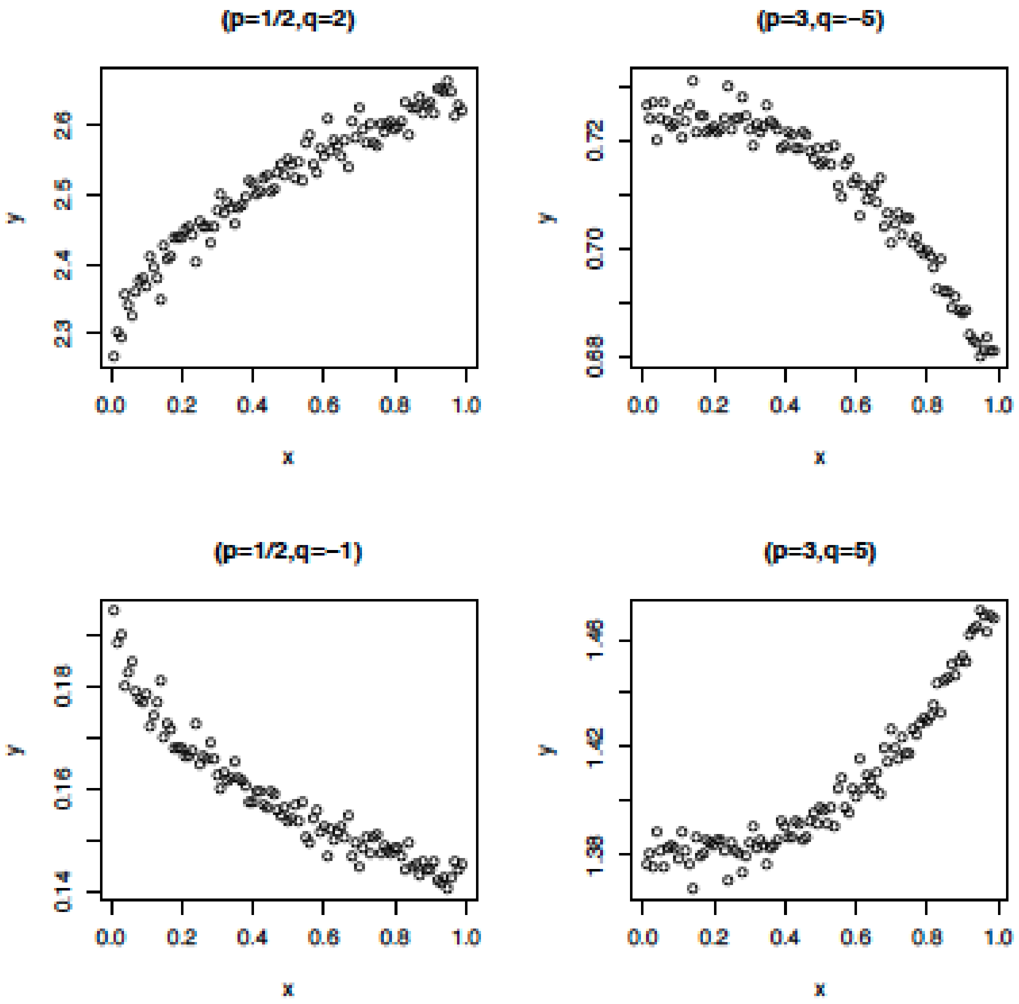
- **Tukey and Mosteller's Bulging Rule**
  - The idea is that it might be interesting to transform x and y at the same time, using some power functions.

$$y_i^q = \beta_0 + \beta_1 x_i^p + \eta_i$$



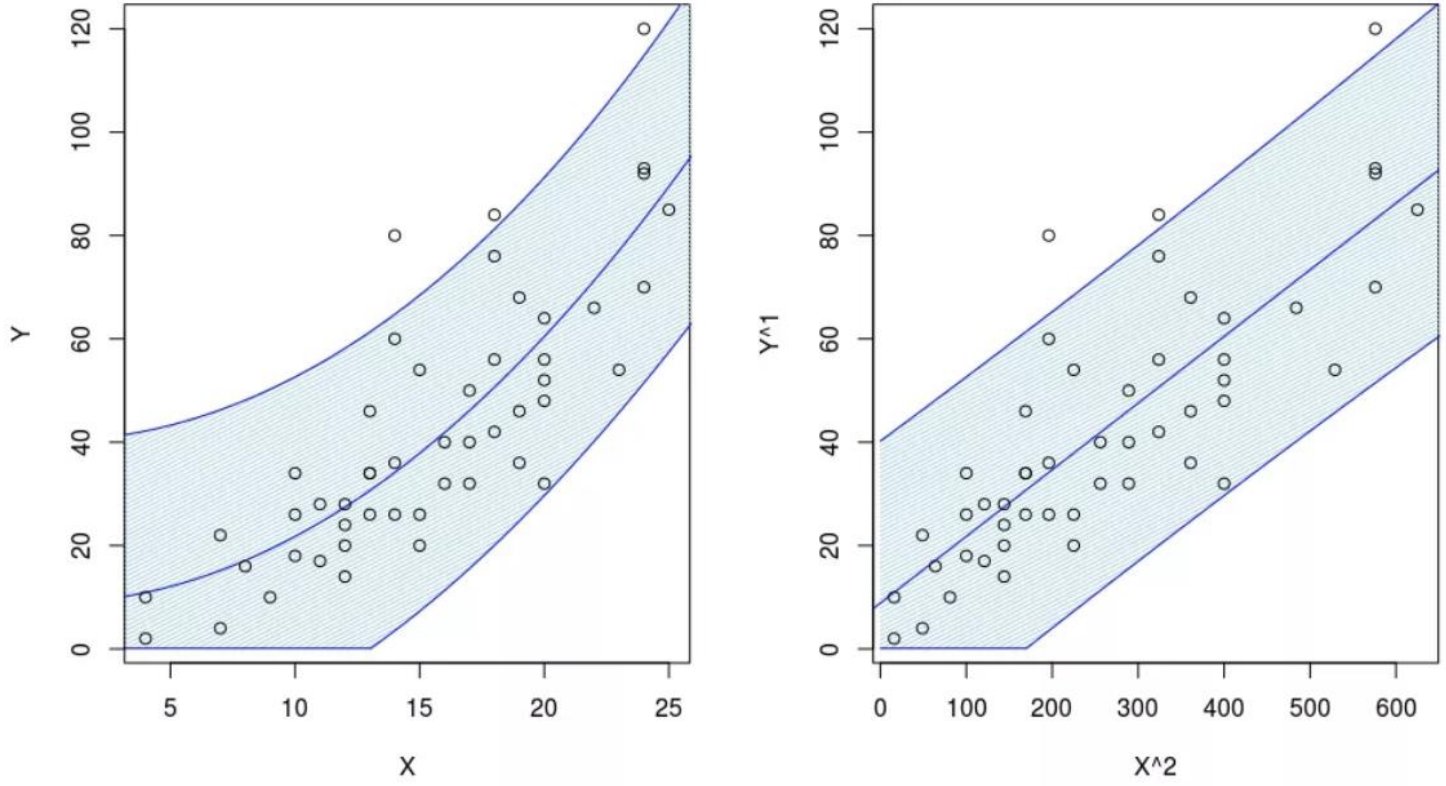
	4	y^4	
	3	y^3	
UP	2	y^2	
	1	y	original data
	1/3	$\sqrt[3]{y}$	cube root transformation
	1/2	$\sqrt{y}$	root transformation
	0	log y	log transformation
	-1/2	$-1/\sqrt{y}$	inverse root transformation
	-1	$-1/y$	reciprocal transformation
Down	-2	$-1/y^2$	
	-3	$-1/y^3$	
	-4	$-1/y^4$	

Power Transformation

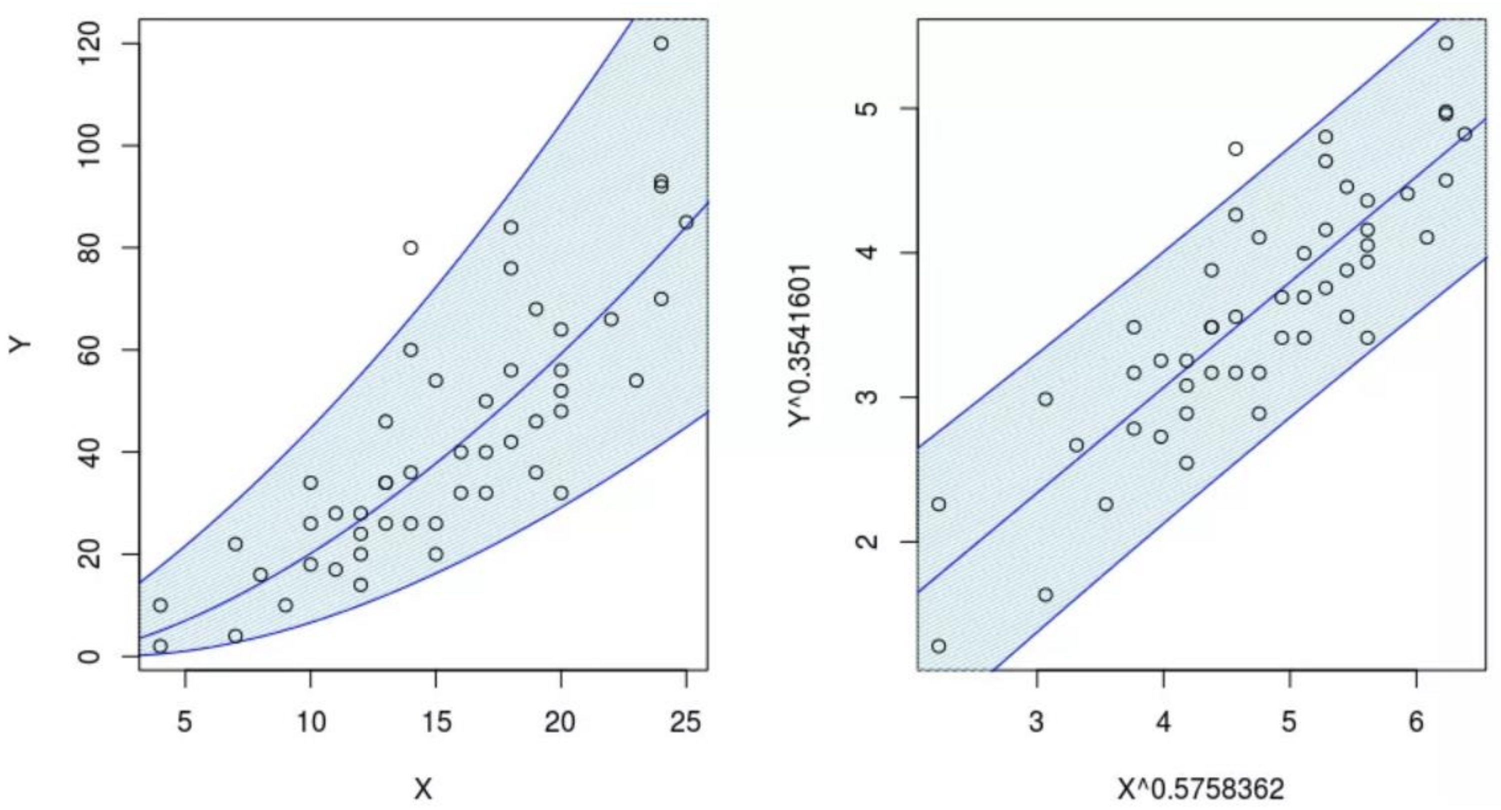


More information can be found https://www.r-bloggers.com/tukey-and-mostellers-bulging-rule-and-ladder-of-powers/

Power Transformation



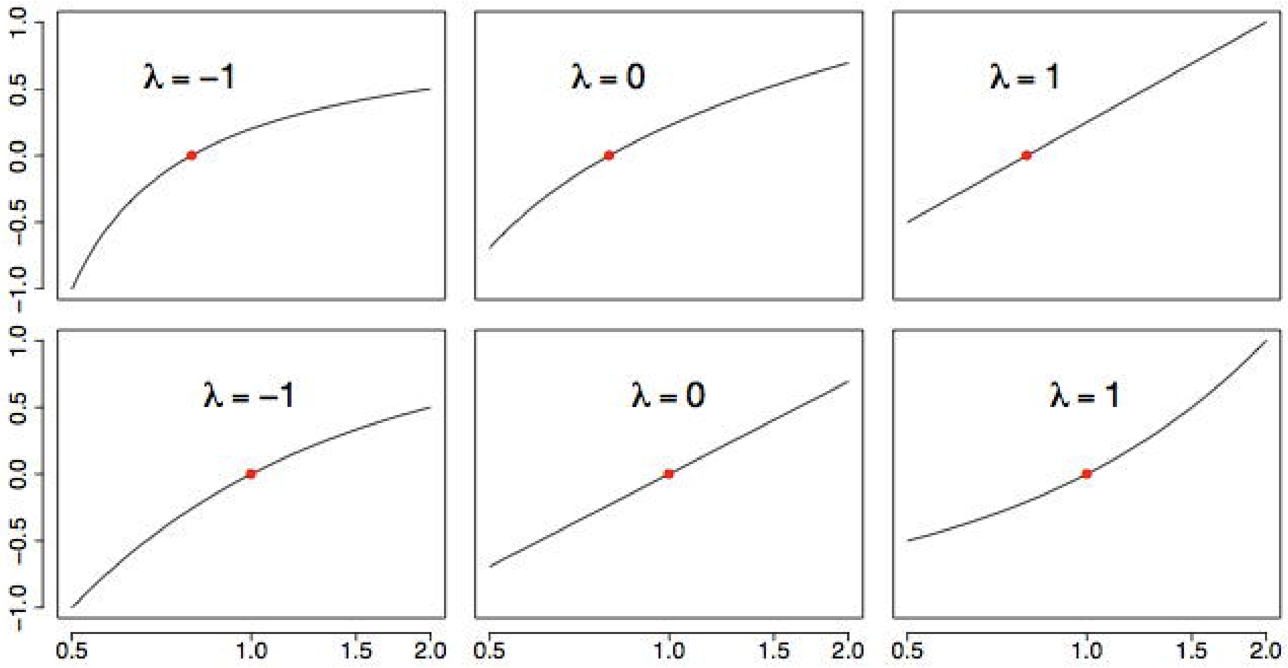
Power Transformation



Power Transformation

- The **Box-Cox Transformation** transforms a continuous variable into an almost normal distribution.

$$y = \begin{cases} \dfrac{x^{\lambda} - 1}{\lambda}, & if\ \lambda \neq 0 \\ \log(x), & if\ \lambda = 0 \end{cases}$$

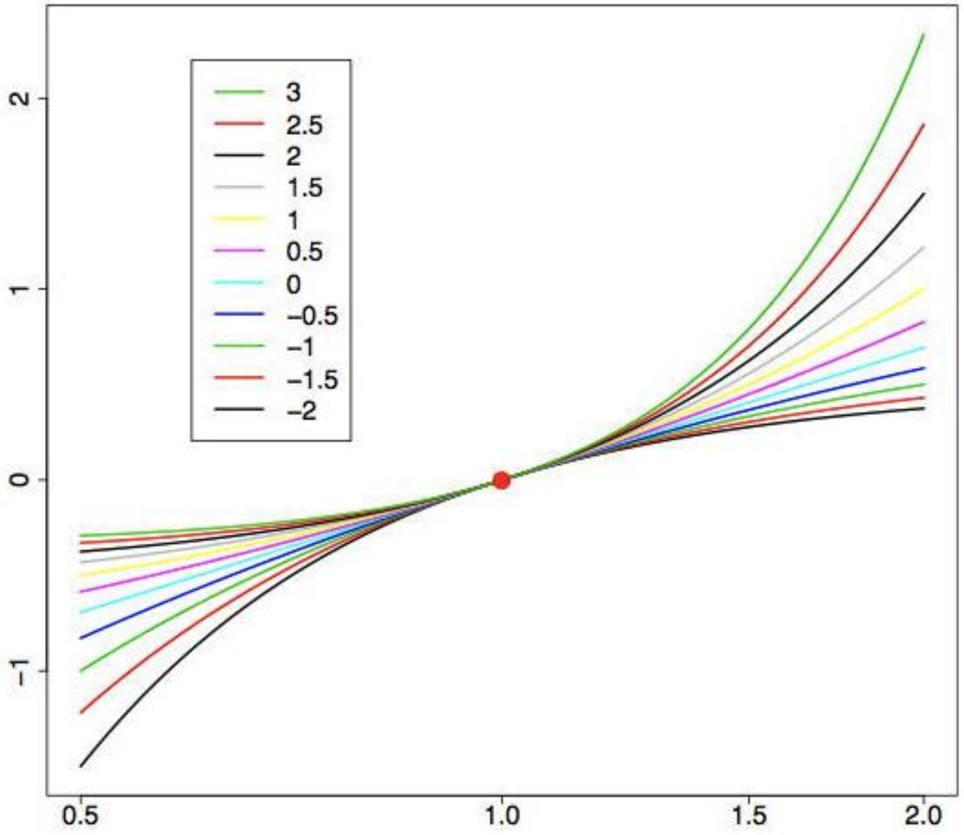


Examples of the Box-Cox transformation $x_{\lambda}{}'$ versus x for $\lambda = -1, 0, 1$. In the second row, $x_{\lambda}{}'$ is plotted against $\log(x)$. The red point is at $(1, 0)$.

Power Transformation

- The **Box-Cox Transformation** transforms a continuous variable into an almost normal distribution.

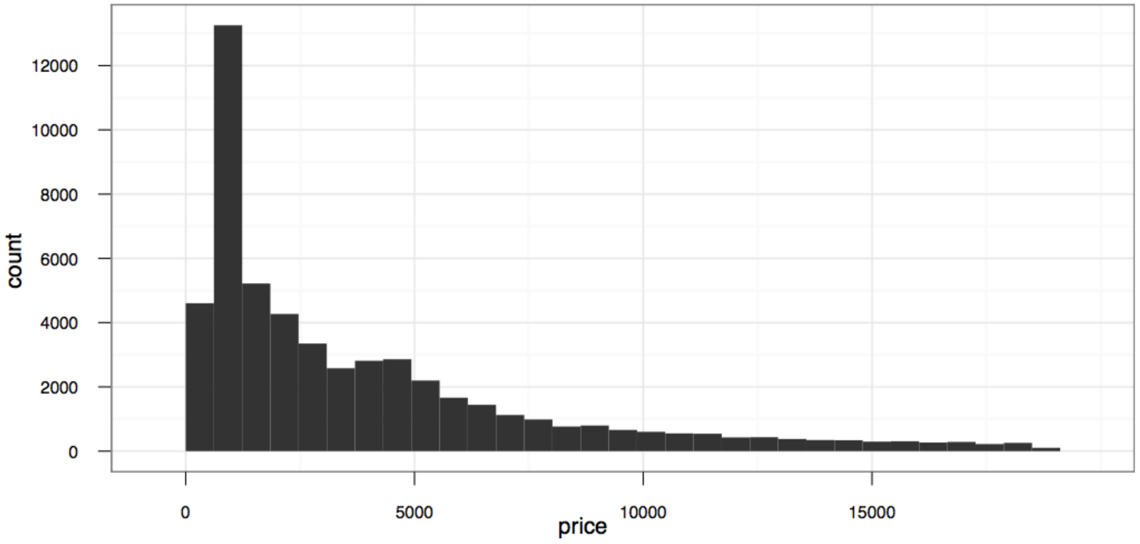
$$y = \begin{cases} \dfrac{x^{\lambda} - 1}{\lambda}, & if\ \lambda \neq 0 \\ \log(x), & if\ \lambda = 0 \end{cases}$$

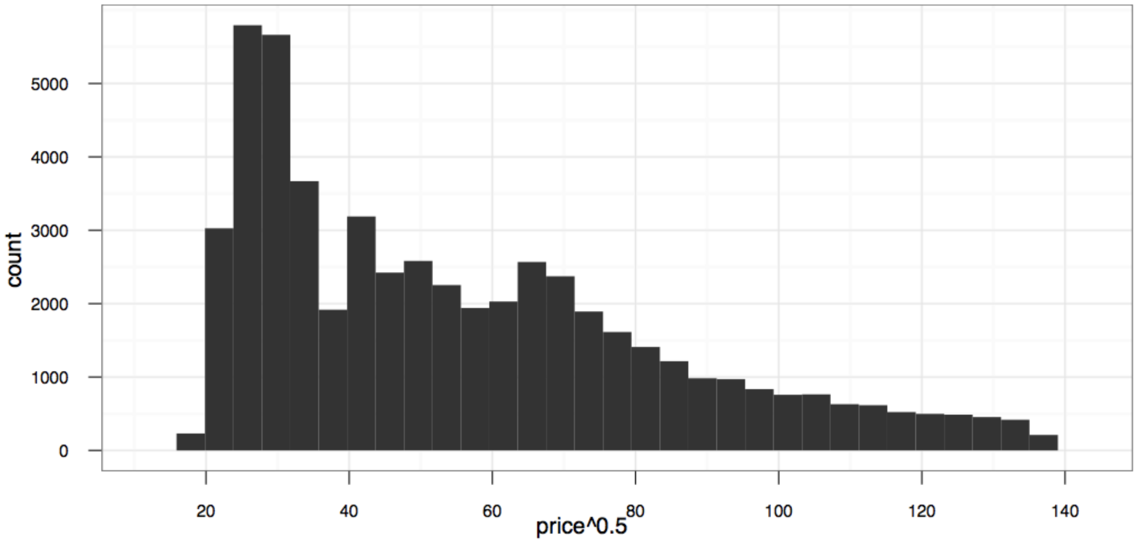


Power Transformation

- The **Box-Cox Transformation** transforms a continuous variable into an almost normal distribution.

$$y = \begin{cases} \dfrac{x^{\lambda} - 1}{\lambda}, & if\ \lambda \neq 0 \\ \log(x), & if\ \lambda = 0 \end{cases}$$

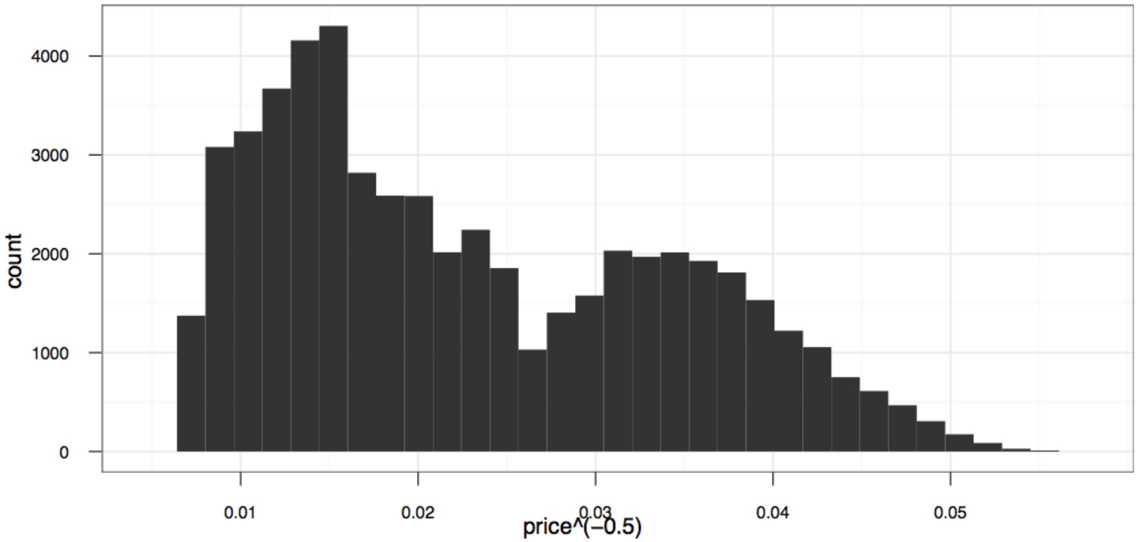




Power Transformation

- The **Box-Cox Transformation** transforms a continuous variable into an almost normal distribution.
  - With negative values in the attributes

$$y = \begin{cases} \dfrac{(x+c)^{\lambda}-1}{g\lambda}, & if\ \lambda \neq 0 \\ \dfrac{\log(x+c)}{g}, & if\ \lambda = 0 \end{cases}$$

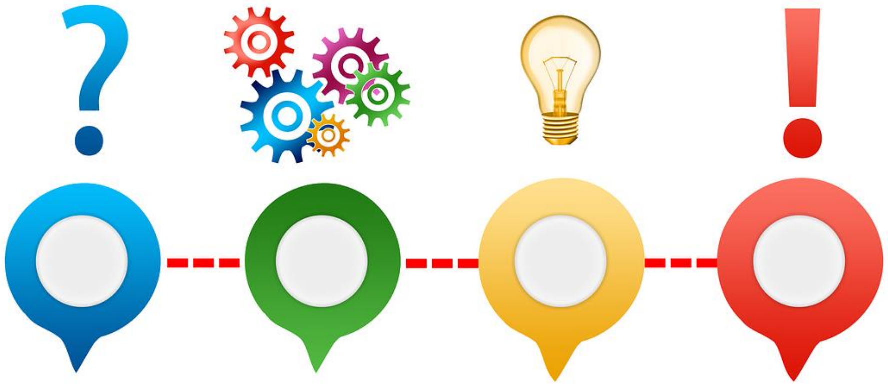


where

- c: offset the negative values.
- g: scale the resulting values, often considered as the geometric mean of the data.
- λ: greedily search λ so that the resulting attribute is as close as possible to the normal distribution.

Data Transformation

- Overview of Data Transformation
- Data Normalisation
- Data Discretisation
- Data Construction
  - Feature Engineering
  - Data Sampling



MONASH University

Data Discretisation

- The process of converting or partitioning continuous variables to discretised or nominal variables.
  - Find concise data representations as categories which are adequate for the learning task retaining as much information in the original continuous attribute as possible
  - Effects of discretisation
    - Smooth data
    - Reduce noise
    - Reduce data size
    - Enable specific methods using nominal data

Binning

- An unsupervised algorithm (doesn't care about the dependent variable) that splits ordered data into predefined number of bins.
- Two approaches
  - Equal-width binning
    - Given a range of values, $[x_{min}, x_{max}]$, we divide the value range into intervals with approximately same width, w

$$w = \frac{x_{max} - x_{min}}{n}$$

where n is the number of bins, or you can specify the value of w

  - Equal-depth binning
    - Divides the range into n intervals, each containing approximately the same number of samples.
- Binning with mean value, median values or bin boundaries

Example - Binning

- Given a set of data: $\{34, 64, 88, 55, 94, 59, 10, 25, 44, 48, 69, 15\}$
  - Sort the values in ascending order

    {10, 15, 25, 34, 44, 48, 55, 59, 64, 69, 88, 94}
  - Equal-width binning with n = 4

    {10, 15, 25}, {34, 44, 48}, {55, 59, 64, 69}, {88, 94}
    - Mean value

      {16.6, 16.6, 16.6}, {42, 42, 42}, {61.75, 61.75, 61.75, 61.75}, {91, 91}
    - Median value

      {15, 15, 15}, {44, 44, 44}, {61.5, 61.5, 61.5, 61.5}, {91, 91}
    - boundaries

      {10, 10, 25}, {34, 48, 48}, {55, 55, 69, 69}, {88, 94}

Example - Binning

- Given a set of data: $\{34, 64, 88, 55, 94, 59, 10, 25, 44, 48, 69, 15\}$
  - Sort the values in ascending order

    {10, 15, 25, 34, 44, 48, 55, 59, 64, 69, 88, 94}
  - Equal-depth binning with n = 4

    {10, 15, 25}, {34, 44, 48}, {55, 59, 64}, {69, 88, 94}
    - Mean value

      {16.6, 16.6, 16.6}, {42, 42, 42}, {59.3, 59.3,59.3,59.3}, {83.6,83.6,83.6}
    - Median value

      {15, 15, 15}, {44, 44, 44}, {59, 59, 59}, {88, $88, 88$}
    - boundaries

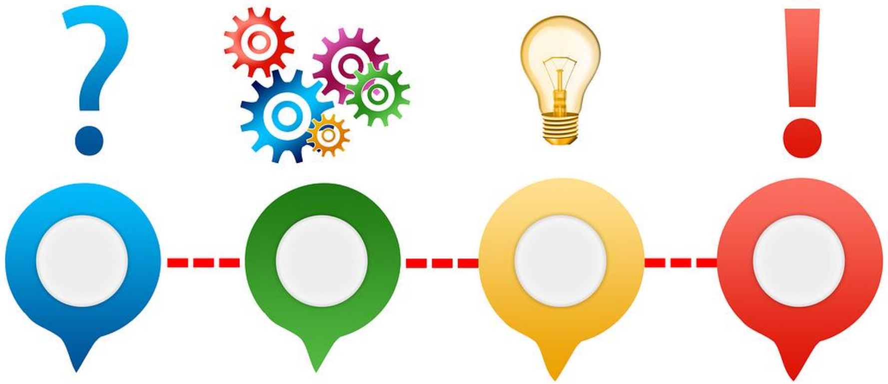
      {10, 10, 25}, {34, 48, 48}, {55, 55, 64}, {69, 94, 94}

Binning

- Advantage/disadvantage of each method:
  - Equal-width binning
    - Is simple but sensitive to outliers
    - Not well handles skewed data
  - Equal-depth binning
    - Scales well by keeping the distribution of the data

Data Transformation

- Overview of Data Transformation
- Data Normalisation
- Data Discretisation
- Data Construction
  - Feature Engineering
  - Data Sampling



Feature Engineering

	Feature Extraction/Generation	Feature Selection
	Generate new features from raw data or other features	Select a subset of available features based on some criteria
Goals	• Produce more meaningful/descriptive/discriminant features	• Remove irrelevant data • Increase predictive accuracy of learned models • Improve learning efficiency • Reduce the model complexity and increase its interpretability

Feature Subset Selection

- Feature subset selection reduces the data set size by removing irrelevant or redundant features.
  - Goal: find a minimum set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes
  - Methods
    - Stepwise forward selection
    - Stepwise backward elimination.
    - Combination of forward selection and backward elimination
    - Decision tree induction.

Example

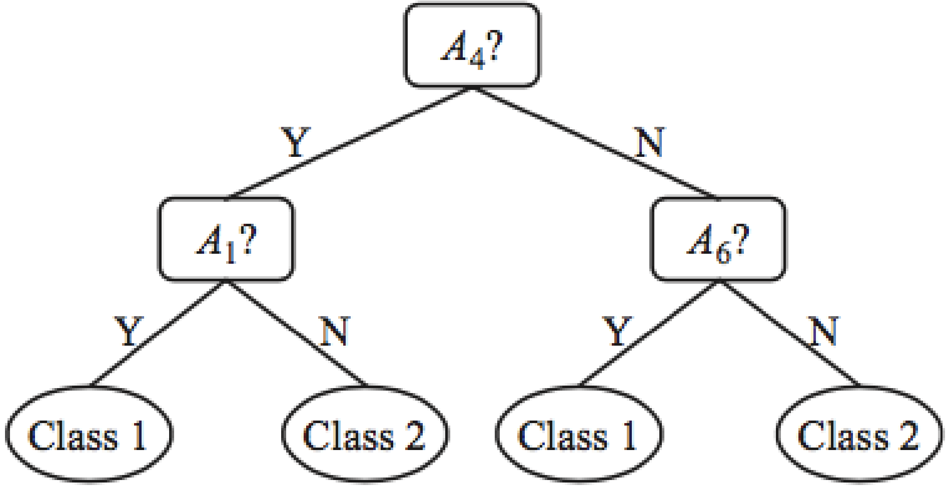
Forward selection	Backward elimination	Decision tree induction
Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ Initial reduced set: $\{\}$ => $\{A_1\}$ => $\{A_1, A_4\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ => $\{A_1, A_3, A_4, A_5, A_6\}$ => $\{A_1, A_4, A_5, A_6\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$  => Reduced attribute set: $\{A_1, A_4, A_6\}$

Figure is from "Data mining: know it all"

Data Sampling

- Sampling methods are used to choose a representative subset of the data
  - Reduce the volume of data
  - Fix imbalance distribution
  - Creating training, validation, testing sets.
- **Methods**: Suppose that a large dataset, D, contains N tuples, the ways we can used to do data reduction:
  - *Simple random sample without replacement* (**SRSWOR**) of size s:
    - Draw s of the N tuples from $D (s < N)$, where the probability of drawing any tuple in D is $1/N$
  - *Simple random sample with replacement* (**SRSWR**) of size s:
    - Similar to SRWOR, except that after a tuple is drawn, it is placed back in D so that it may be drawn again.

Data Sampling

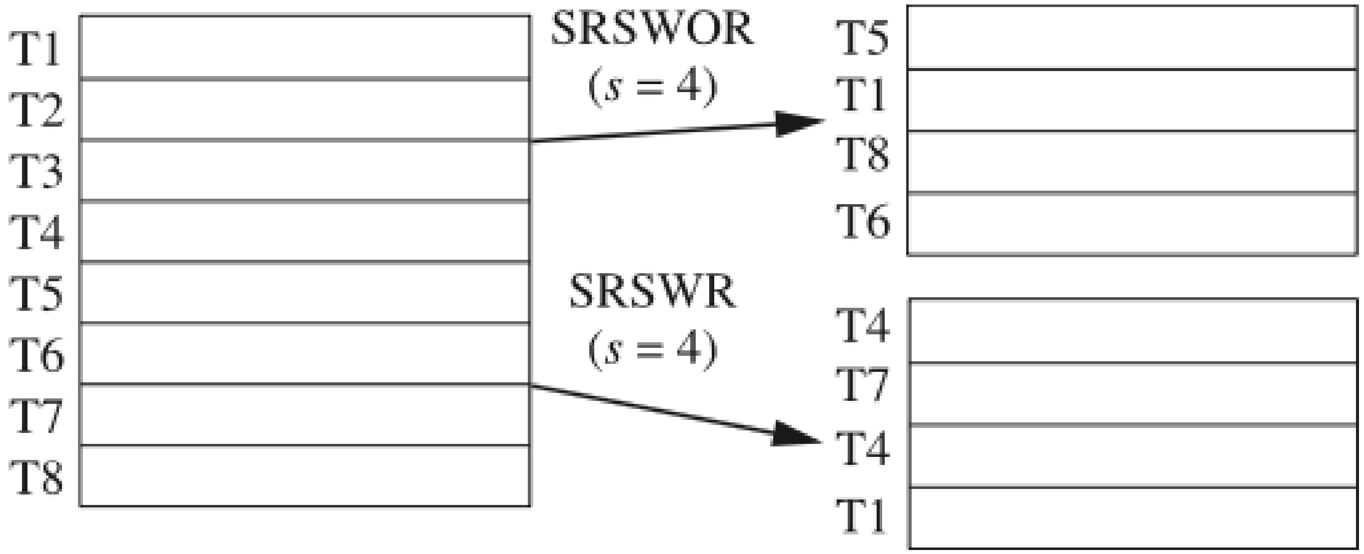


Figure is from "Data mining: know it all"

Data Sampling

- **Methods**: Suppose that a large dataset, D, contains N tuples, the ways we can used to do data reduction:
  - Stratified sample:
    - If D is divided into mutually disjoint parts called strata, a stratified sample of D is generated by obtaining an SRS at each stratum

T38	youth
T256	youth
T307	youth
T391	youth
T96	middle_aged
T117	middle_aged
T138	middle_aged
T263	middle_aged
T290	middle_aged
T308	middle_aged
T326	middle_aged
T387	middle_aged
T69	senior
T284	senior

T38	youth
T391	youth
T117	middle_aged
T138	middle_aged
T290	middle_aged
T326	middle_aged
T69	senior

Figure is from "Data mining: know it all"

Summary

- Please download and read materials provided on Moodle.
- Review content learnt from Week 8.

- Assessments
  - Complete your group selection for Assessment 2
  - Read the tasks in Assessment 2 and work on them.

- Next week: Enjoy your break!!!