





Exploratory Data Analysis

 Files	
 Notes	
 Status	
 Topics	

Process

▼ Steps:

- 1. Generate questions about our data.
- 2. Search for answers by visualising, transforming and modeling our data.
- 3. Use what we learn to rene our questions and/or generate new questions.

▼ Note:

- We should not expect to have the final plot ready in an instant. Even if we know exactly what kind of plot we want (e.g. scatterplot, bar chart, etc.) we should expect to plot it several times over.
- Each time we plot it, we shall vary the colours, titles, labels, and so on, until we are satisfied that it highlights the data and the message that it carries with it.

▼ Definitions:

- A **variable** is a quantity that we measure. In our data, it should be stored in its own column.
- A **value** is the state of a variable when we measure it. It should be in a single cell in our table.
- An **observation** is a set of measurements made under similar conditions. For data to be tidy, each observation should be stored in its own row.
- A variable is **categorical** if it can only take on a small set of values. In R, these are typically stored as a factor.
- A variable is **continuous** if it can take on an infinite set of ordered values. In R, these are typically stored as numeric or integer.

Association Measures

▼ Definition

- 2 Categorical variables are:
 - **Independent** if the conditional proportions for one of them are identical at each category of the other
 - **Dependent** or **associated**, if the conditional proportions are not identical
- Knowing the value for one the variables gives us additional information about the other one
 - For e.g. If we know that the clarity of a diamond is high, we could postulate that the cut is lower since there are negatively associated

▼ Difference in Proportions

▼ Formula

$$\hat{p}_1 - \hat{p}_2$$

- Note that \hat{p}_1, \hat{p}_2 are the proportions of each of the people in the different subgroups

▼ Range:



-1 to 1

- -1 : Lower bound, Highest Association
- 0 : Middle, No Association
- 1 : Upper bound, Highest Association
- Note that the strength of the association does not change for either order of the difference that we take, only the sign changes

▼ Relative Risk

▼ Formula

$$\frac{\hat{p}_1}{\hat{p}_2}$$

- Note that \hat{p}_1, \hat{p}_2 are the proportions of each of the people in the different subgroups

▼ Range:



0 to ∞

- 0 : Lower bound, Highest Association
- 1 : Middle, No Association
- ∞ : Upper bound, Highest Association

▼ Use Case:

- Preferable to use when both of the proportions are close to 0 so that we can have a better gauge of what is the difference in the proportions

▼ Log Scale

- Note that we can change it to a log scale as well so that the range will be symmetric about 1 and it will go from $-\infty, \infty$

▼ Odds Ratio

▼ Odds

▼ Formula

$$\text{odds} = \frac{p}{1 - p}$$

- Note that p is the probability of success and $1 - p$ is the probability of failure
- This is also the odds of success

▼ Range



0 to ∞

- Odds equal to 0 corresponds to probability of success equal to 0.
- Odds equal to 1 corresponds to probability of success equal to 0.5.
- Odds equal to ∞ corresponds to probability of success equal to 1.

▼ Log Scale:

- Log-odds equal to $-\infty$ corresponds to probability of success equal to 0.
- Log-odds equal to 0 corresponds to probability of success equal to 0.5.
- Log-odds equal to ∞ corresponds to probability of success equal to 1.

▼ Formula

$$\frac{\hat{p}_1}{1 - \hat{p}_1} / \frac{\hat{p}_2}{1 - \hat{p}_2}$$

- Note that \hat{p}_1, \hat{p}_2 are the probability of success for each of the 2 subgroups
- This is the ratio between the odds of success between two groups

a	b
c	d

$$\frac{ad}{bc}$$

- Note that we can also express the odds ratio as such using the numerical values

▼ Range:

💡 $-\infty$ to ∞

- $-\infty$: Lower bound, Highest Association
- 0 : Middle, No Association
- ∞ : Upper bound, Highest Association

▼ Use Case:

- Best used when computing the proportions from the way we collect the data is useful

▼ Important Note

- The odds ratio is the **same no matter which direction** that we take it which is very useful when we are doing retrospective or prospective studies
 - When we collect data, we are limited to the way in which we are allowed to compute the data. We can only compute the proportions row wise or column wise. By using odds ratio, it will be the same either ways and it will show us whether there is association between the variables
- Note that the odds ratio is **indifferent to scaling difference**. i.e. If there are multiplicative amounts row wise or column wise
 - We could collect more data for a certain row or column but it will be indifferent to the scaling difference for those
- If $\hat{p}_1, \hat{p}_2 \approx 0$, then **Odds Ratio \approx Relative Risk**

$$\begin{aligned} OR &= \frac{\frac{\hat{p}_1}{1 - \hat{p}_1}}{\frac{\hat{p}_2}{1 - \hat{p}_2}} \\ &= \frac{\hat{p}_1}{\hat{p}_2} \left(\frac{1 - \hat{p}_2}{1 - \hat{p}_1} \right) = RR \times \left(\frac{1 - \hat{p}_2}{1 - \hat{p}_1} \right) \\ \text{If } \hat{p}_1, \hat{p}_2 \approx 0, & \text{ then } OR \approx RR \\ &= \end{aligned}$$

▼ Log Scale

- Note that we can also make use of the log scale as well, we can compute the log-odds and we compare the ratio between the log-odds

▼ Concordant and Discordant Pairs

- It is also another form of association analysis

▼ Steps

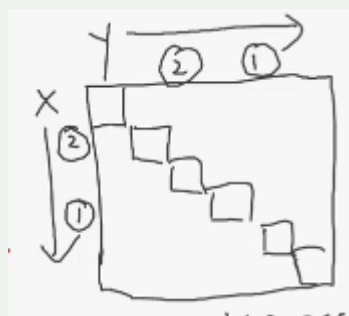
- Suppose that we have 2 Ordinal Variables X and Y

- We take 2 random samples and we check their X and Y values and check if they are concordant or discordant pairs
- We will repeat this for a few samples and see what is the overall result and where our values lie

▼ Types

Suppose that we take 2 samples from the dataset and record the value of the variables

- **Concordant Pairs:**
 - If for any variable, sample X has a larger value than Y, then the value for the variable of X will also be larger
 - The signs for the comparison between the values of the variables of the 2 samples will be the **same**
 - $(X_1 > X_2) \& (Y_1 > Y_2)$
 - $(X_1 < X_2) \& (Y_1 < Y_2)$
 - **Extreme Case:**
 - If all the values falls along this diagonal and we know that they are concordant pairs, then we also know that there is a **positive association**



- **Discordant Pairs:**
 - If for any variable, sample X has a larger value than Y, then the value for the variable of X will be smaller instead
 - The signs for the comparison between the values of the variables of the 2 samples will be **different**
 - $(X_1 > X_2) \& (Y_1 < Y_2)$
 - $(X_1 < X_2) \& (Y_1 > Y_2)$
 - **Extreme Case:**
 - If all the values falls along this diagonal and we know that they are discordant pairs, then we also know that there is a **negative association**



Contingency Tables

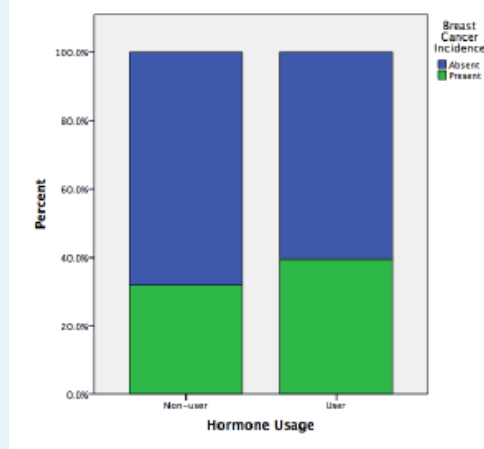
▼ Definition

- Display for 2 categorical variables
- Rows and Columns lists out the categories of each of the variables
- Entry in each of the cells of the table is the number of observations in the sample at a particular combination of categories of the 2 variables
- Note that we can have any number of categories along the rows and columns and it will become a $i \times j$ contingency table
- We can make use of `count()` and `pivot_wider` to create a contingency table

▼ Visualisations

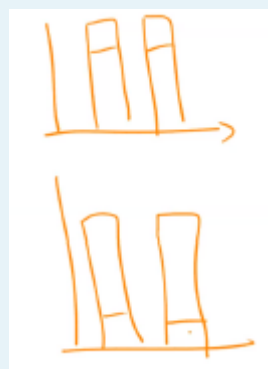
▼ Filled Stacked Bar Charts

- Use `geom_bar(position = "fill")` so that we can see the proportions



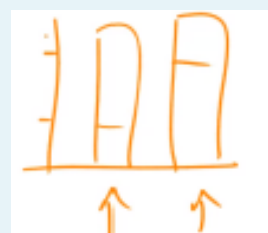
A stacked bar chart provides us with a visual representation of the difference in the proportion

- **Low Association:**



When the different groups have the same proportion across the different variables

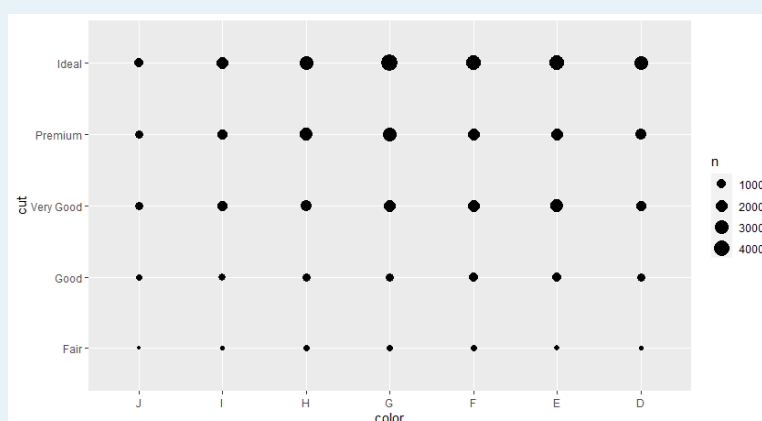
- **High Association:**



When the different groups have varying proportion across the different variables

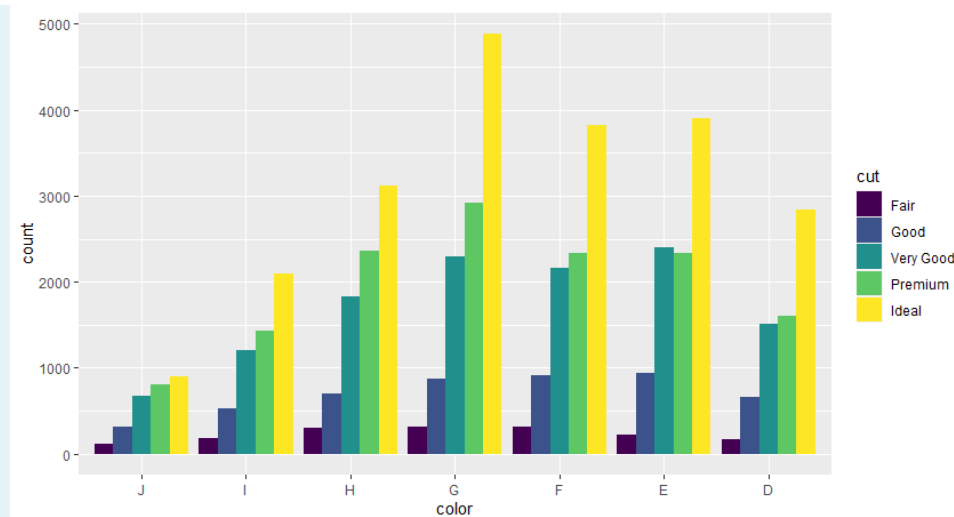
▼ Size based on the counts

- Make use of `geom_count()`



▼ Dodged Bar Charts

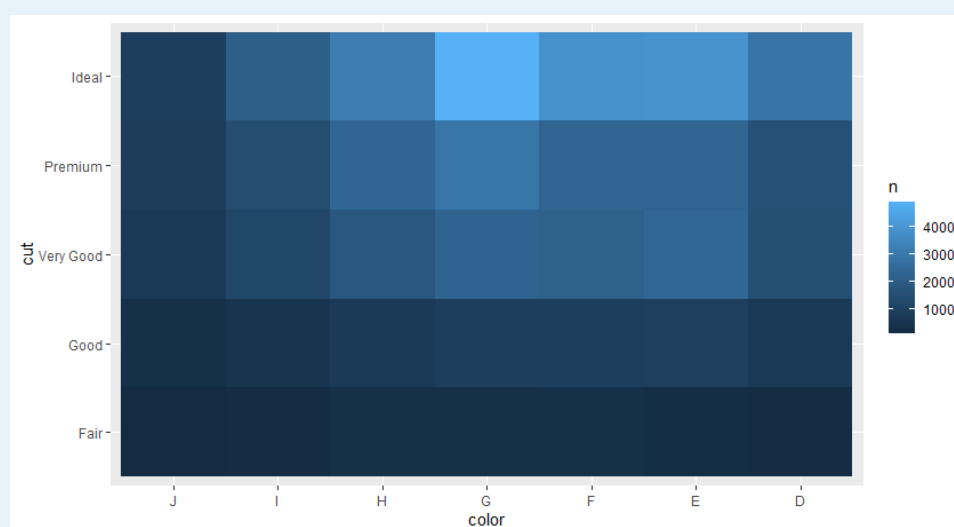
- Using `geom_bar(position = "dodge")`



We can compare each of the distributions at the same x location

▼ Heat Maps

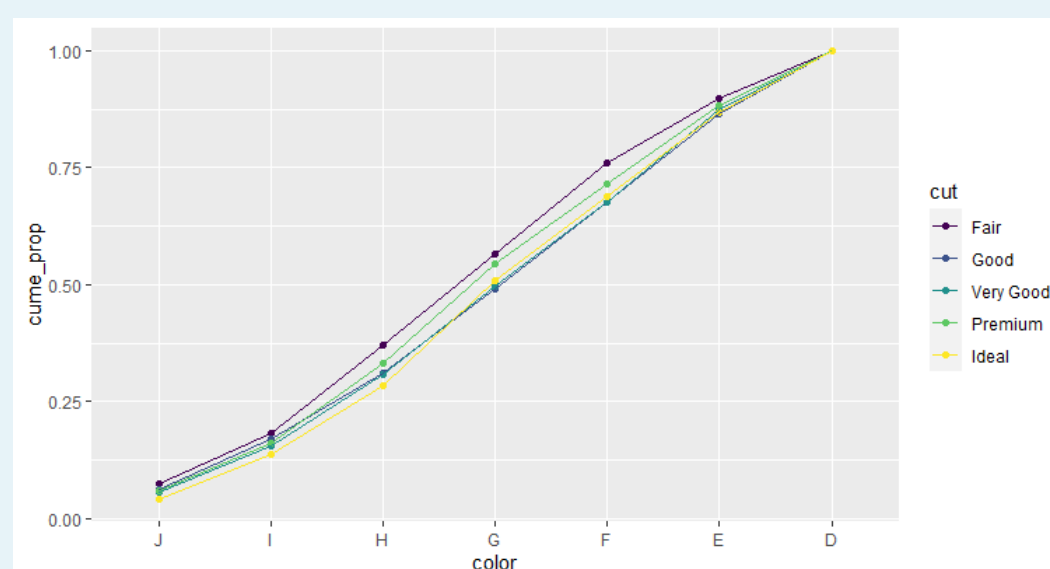
- Make use of `geom_tile()`



▼ Cumulative Distribution Graphs

▼ Steps

- Make use of `cumsum()` to compute the cumulative sum
- Make use of `cumsum()` and `sum()` to compute the cumulative proportion
- Make use of `geom_point()` and `geom_line()` to plot
- We can compare cumulative distribution to see if there are any trends and if there are any particular group that is always higher than the rest
- Good when we have ordinal variables



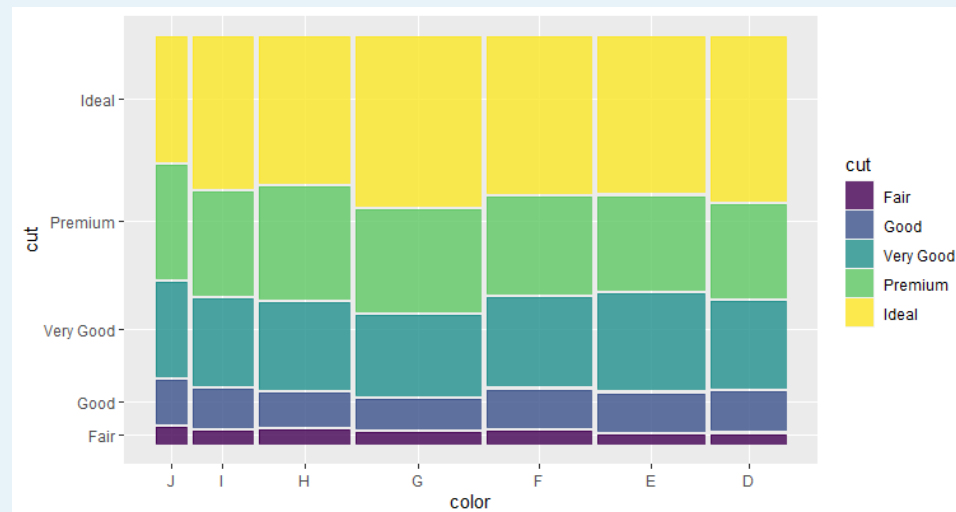
Example Code

```
group_by(d2, color, cut) %>% # group by colour and cut
count() %>% arrange(cut, color, .by_group = TRUE) %>% # we count the colour and cut
group_by(cut) %>% # group by only cut
mutate(nn = cumsum(n), # compute the cumulative sum for each of the cuts
       cume_prop = nn/sum(n)) %>% # compute the proportion which is the cumulative sum / total sum
```

```
ggplot(aes(x = color, colour = cut, fill = cut, y = cume_prop, group = cut)) +
  geom_point() + geom_line()
```

▼ Mosaic Charts

- It is similar to stacked bar charts, but the size of the rectangles will represent the counts instead of proportions. The width can tell us about the marginal counts of each of the groups
- Under `ggmosaic` library, use `geom_mosaic()`
 - Note that the `x` aesthetic needs to be used with `product(variable_name)`



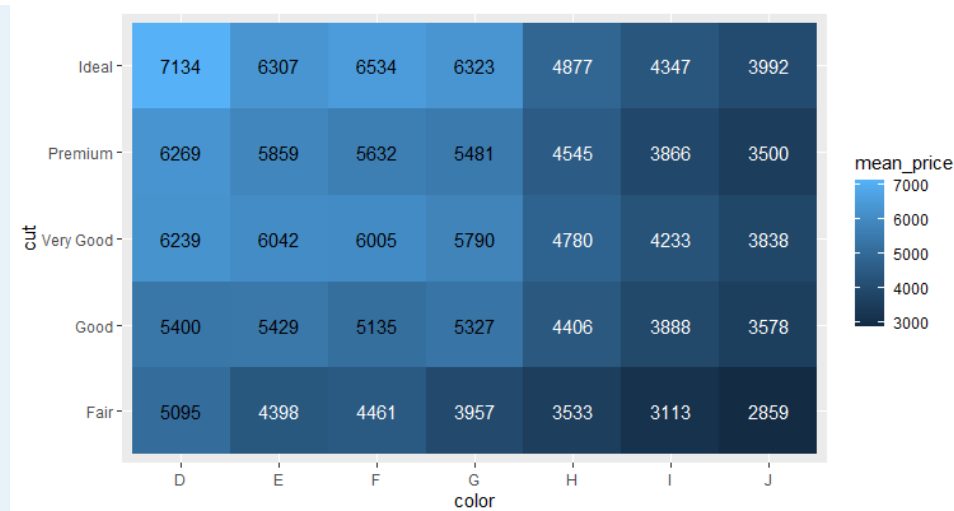
Example of Mosaic Chart

▼ Tables

- Make use of `ggtable()` under `GGally`
- `ggtable()`
 - `cells` - We can specify the type of data that we want to be represented under each of the cells
 - `col.prop` - Column Proportion
 - `row.prop` - Row Proportion
 - `prop` - Overall Proportion
 - `std.resid` - Standardised Residuals (Computed through Chi Squared Test)
 - `resid` - Residuals (Computed through Chi Squared Test)
 - `fill` - We can choose what we want to colour the values with
 - `std.resid` - Red will be that the values are lesser than the expected value, White is not much difference, Blue is that the values are greater than the expected values
 - `resid` - Red will be that the values are lesser than the expected value, White is not much difference, Blue is that the values are greater than the expected values

▼ Pivot Tables

- We will have a table where the row and columns are **categorical variables** and for each of the cells within it, it will be a third variable which will be a form of statistical summary of the 2 factors
- We can make use of `geom_tile` for the creation of the grid and `geom_text` for the annotation of the third variable value



Each of the cells here are the mean of the combination for the color and cut

- Note that we can add in a new column for our data to group values that will hit a certain colour range. Then we can make use of `scale_color_manual` to change the colours. The criteria for the colour change is dependent on which value we think we need to change the text colour to increase readability

```
# data preparation
price_table <- diamonds %>% filter(between(carat, 0.95, 1.05)) %>% # filtering those with carat between 0.95 and 1.05
  group_by(cut, color) %>%
  summarise(mean_price = mean(price), .groups="drop") %>% # find the mean values for the cut and color combination
  mutate(text_col = if_else(mean_price <= 4900, "white", "black")) # we give it a value of either white or black and the value
  is dependant on our own judgement (added column)

# plotting
ggplot(price_table) +
  geom_tile(aes(x=color, y=cut, fill=mean_price)) +
  geom_text(aes(x=color, y=cut, label=round(mean_price,0), color=text_col), show.legend = FALSE) +
  scale_color_manual(values=c("white"="white", "black"="black"))# changing colours manually
```

▼ Analysis

▼ Package

- `DescTools` - There are many functions in this library that allows us to describe the data

▼ Function

`Desc()`

- Note that we will need to pass in the data as a table into the function
- `plotit` - logical value. Whether a plot should be plotted
- `verbose` - `c(1, 2, 3)` Ranked from least to most results. Default value is 2

▼ Tests

▼ Goodman Kruskal γ

▼ Usage

- Can be used when we have **Ordinal** against **Ordinal** variables
- Note that association is symmetric because even if we flip the variables to be either rows or columns, the γ value will still be the same
- This is more general use case as compared to **Somer's D** since it does not take into account which is the independent or dependent variable

▼ Comparisons


- When we compare against other $i \times j$ tables, we could have the same Y variable and we want to see which X variable has a higher association. We can look at the magnitude of the γ to see which one has a stronger association

▼ Formula

$$\gamma = \frac{P(\text{Concordant}) - P(\text{Discordant})}{P(\text{No X or Y Tie})}$$

- Note that the idea of Goodman Kruskal follows from the idea of **Concordant and Discordant Pairs**
- This is basically a conditional probability stating that given there is no tie for the X or Y values, what is the difference in the probability of concordant and probability of discordant

▼ Range

 $-1 \leq \gamma \leq 1$

- 1 - Positive Association
- 0 - No Association
- -1 - Negative Association

▼ Somer's D

▼ Usage

- Both our variables are **Ordinal** variables
- However, we have one **dependent** variable and one **independent** variable
- Note that there is:
 - **Somer's D R|C** - Where it is the probability of the rows given the columns. Note that the columns are the independent variables here and the rows are the dependent variables
 - **Somer's D C|R** - Where it is the probability of the columns given the rows. Note that the rows are the independent variables here and the columns are the dependent variables
- Note that the association is **asymmetric** and it depends on the way that we order the rows and the columns since we will need to read off the one that is the independent variable

▼ Comparisons

- When we compare against other $i \times j$ tables, we could have the same Y variable and we want to see which X variable has a higher association. We can look at the magnitude of the γ to see which one has a stronger association


▼ Formula

$$D_{XY} = \frac{P(\text{Concordant}) - P(\text{Discordant})}{P(\text{No X or Y Tie in Dependent})}$$

$$= \frac{N_C - N_D}{N_C + N_D + N_T}$$

- N_C - Number of Concordant Pairs
- N_D - Number of Discordant Pairs
- N_T - Number of neither concordant nor discordant pairs that are tied on the independent variable but not on the dependent one
- Note that the idea of Somer's D follows from the idea of **Concordant and Discordant Pairs**

▼ Range

 $-1 \leq D_{XY} \leq 1$

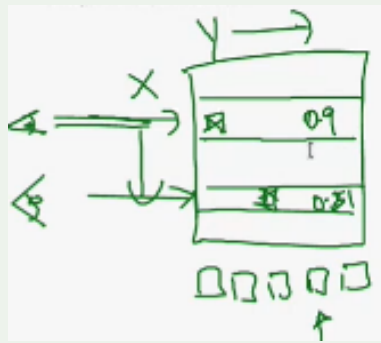
- 1 - Positive Association
- 0 - No Association
- -1 - Negative Association

▼ Goodman Kruskal λ

▼ Usage

- The variables X and Y can be **either nominal or ordinal**
- **Main Idea:** How much extra information we can get from knowing the value of X and predicting Y and not knowing X and predicting Y
 - We take X as the independent and Y as the dependent, but it can be any of it so long as we know whether our row or our column is the dependent or independent variable
- Note that there is:
 - **Lambda R|C** - Where it is the probability of the rows given the columns. Note that the columns are the independent variables here and the rows are the dependent variables
 - **Lambda C|R** - Where it is the probability of the columns given the rows. Note that the rows are the independent variables here and the columns are the dependent variables

▼ Steps



- We compute the marginal proportions for the Y variable and we will predict the Y category to be the modal category
- Then we look at each of the values of X and see what is the new modal category. If the modal category changes if we know X then we will have more information when we know what is X and it will increase our prediction accuracy
- λ computes how much our prediction accuracy increases by the virtue of knowing the value of X

▼ Note

- This is a rather blunt instrument because it will only check if there is a difference in the modal categories. If the modal categories are the same, they will not check the difference in proportion and it will report that there is 0 association
- It could be quite normal to see that the association value is 0

▼ Formula

$$\lambda = \frac{V(Y) - E(V(Y|X))}{V(Y)}$$

- $V(Y)$ - Some measure of variability of the Y value (Could be prediction error)
- $E(V(Y|X))$ - Expected change in the variability of Y given the value of X
- This formula is looking at the difference in the expected variability of Y given some value of X

▼ Range

💡 $0 \leq \lambda \leq 1$

- 0 - No association (Knowing the value of X does not give us any additional information)
- 1 - Strong association

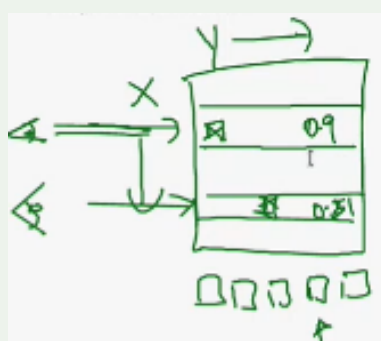
▼ Uncertainty Coefficient

▼ Usage

- The variables X and Y can be **either nominal or ordinal**
- Difference with **Goodman Kruskal λ** is that this uses Entropy instead for the variability method of computation

- **Main Idea:** How much extra information we can get from knowing the value of X and predicting Y and not knowing X and predicting Y
 - We take X as the independent and Y as the dependent, but it can be any of it so long as we know whether our row or our column is the dependent or independent variable
- Note that there is:
 - **Uncertainty Coefficient R|C** - Where it is the probability of the rows given the columns. Note that the columns are the independent variables here and the rows are the dependent variables
 - **Uncertainty Coefficient C|R** - Where it is the probability of the columns given the rows. Note that the rows are the independent variables here and the columns are the dependent variables
 - **Uncertainty Coefficient sym** - Where we take some weighted average of the values that are given by the 2 above values where we take either X or Y to be independent

▼ Steps



- We compute the marginal proportions for the Y variable and we will predict the Y category to be the modal category
- Then we look at each of the values of X and see what is the new modal category. If the modal category changes if we know X then we will have more information when we know what is X and it will increase our prediction accuracy
- **Uncertainty** computes how much our prediction accuracy increases by the virtue of knowing the value of X

▼ Note

- This is a rather blunt instrument because it will only check if there is a difference in the modal categories. If the modal categories are the same, they will not check the difference in proportion and it will report that there is 0 association
- It could be quite normal to see that the association value is 0

▼ Formula

$$\text{Uncertainty} = \frac{V(Y) - E(V(Y|X))}{V(Y)}$$

- $V(Y)$ - Some measure of variability of the Y value (Entropy is used for this)
- $E(V(Y|X))$ - Expected change in the entropy of Y given the value of X
- This formula is looking at the difference in the expected entropy of Y given some value of X

▼ Range

💡 $0 \leq \text{Uncertainty} \leq 1$

- 0 - No association (Knowing the value of X does not give us any additional information)
- 1 - Strong association

▼ Mutual Information

- Measurement between the difference in the product of marginal distribution and the joint distribution

Colour Palettes

▼ Definition

- Colour Palette is a range of colours
- Can be used to distinguish between groups of data
- We can use the values for continuous data by binning them and different bins will have different colours

▼ Palettes

- Under `library(RColorBrewer)`
- `display.brewer.all()` - Displays all the palettes that we can use

▼ Functions

- Return the hexadecimal color specification - `brewer.pal(n, name)`
- Display a single RColorBrewer palette by specifying its name - `display.brewer.pal(n, name)`
- Display all color palette - `display.brewer.all(n = NULL, type = "all", select = NULL, colorblindFriendly = FALSE)`
 - `n` : Number of different colors in the palette, minimum 3, maximum depending on palette.
 - `name` : A palette name from the lists above. For example `name = RdBu` .
 - `type` : The type of palette to display. Allowed values are one of: “div”, “qual”, “seq”, or “all”.
 - `select` : A list of palette names to display.
 - `colorblindFriendly` : if TRUE, display only colorblind friendly palettes.

▼ Concept of Palettes:

Three-Dimensional Concept

- **Hue**: E.g. Red, Green Blue
 - Change of hue can perceive differences
- **Value**: Light vs Dark
 - Change of lightness can aid in perceiving ordering
- **Saturation**: Dull vs Vivid

▼ Set of Color Palettes



1. **Top:** Sequential Colour Palettes where it goes from light to dark
 - a. Can be used to show ordering of values
 - b. When we want our reader to identify which values are higher or lower than other values
2. **Middle:** Unordered Categorical Variables. No ordering and it is a mix match
3. **Bottom:** Divergent Colour Palette. Goes from Dark to Faint to Dark again
 - a. When we want our viewer to identify ordering in two directions
 - i. For example, we can assign blue to regions that have lower than average rates and red to regions having higher than average rates
 - ii. Varying the lightness of each of the hues also allows the readers to order the regions
 - b. If there is a middle point and we want to separate into 2 different regions with the regions being ordered as well

Correlation Matrix

▼ Definition

- Matrix that represents the pair-wise distance between different variables
- `cor(., use = "pair")` - We can pass our data into the function so that it computes all the pairwise correlations and give us the correlation matrix

▼ Hierarchical Clustering

▼ Definition

- Form of unsupervised learning technique (Try to find a structure to the data)
- Find groups/clusters in our data (where we want to find members within a group that are "similar" to one another)
- The output of a hierarchical clustering allows for visualisation of N-1 possible clustering at one go, using a dendrogram

▼ Alternatives

- K-means Clustering

▼ Usage

- After we get the clustering, we could just choose one of the variables from each of the groups instead of including all the variables.
- It can help to reduce the dimensionality and aid in feature selection
- This can help to reduce the noise in our data and reduce multicollinearity problems
- When we want to plot the heatmap for the correlation matrix, we can group those that are more similar to each other together

▼ Input

Suppose that we have **N observations** $x_1, x_2 \dots, x_N$ where we want to group them into **K clusters**. Each observation is typically a vector of p measurements

- $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ can be seen as the measurements that we have for each of the observations
- Note that we can treat the columns of our data set as the observations if we want to cluster the variables together. In this case, the measurements for the variables are the different rows (values) that we
- **Dissimilarity Matrix** (We only need this)

▼ Dissimilarity Measures

- Note that for different problems we will need to think of what is the appropriate Dissimilarity Measure to use

▼ Between Individual Observations

Let d be a binary function of two observations, that measures pairwise dissimilarity

- **Euclidean Distance**
 - One of the most common choice

$$d(x_i, x_j) = \sum_{s=1}^p (x_{is} - x_{js})^2$$

Formula

- **$L1 - norm$**
 - Also known as the Manhattan Distance

$$d(x_i, x_j) = \sum_{s=1}^p |x_{is} - x_{js}|$$

Formula

- **Jaccard Index**
 - Can be used when we have categorical data
- **Gower's Distance**
 - Can be used when we have a mixture of categorical and numerical data

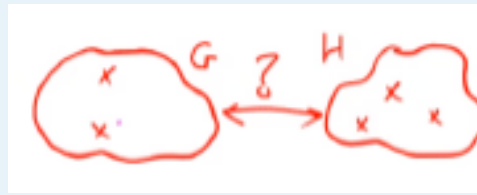
▼ Between Groups

- To build on the pairwise dissimilarity to obtain a measure of dissimilarity between groups
- Suppose that we have 2 groups, G and H with N_G and N_H points within them respectively and we will want to find a way to compare the dissimilarity between the groups

▼ Linkage Method

- Computes the dissimilarity between groups, using only point-wise dissimilarities

- Note that we will look at the pairwise distances between all the points from the 2 different groups, where we will have $N_G \times N_H$ pairs of distances we can compare and we will use one of the linkage methods to compute the dissimilarity between groups



We will find the distances between the points of group G and H

▼ Types

When we are starting out, we can make use of **Average Linkage** or **Ward Linkage**. Unless we know what type of data we have, we can make use of **Single Linkage** or **Complete Linkage**

- Plot the data to see the structure of it then we do the clustering

Comparing different hierarchical linkage methods on toy datasets

This example shows characteristics of different linkage methods for hierarchical clustering on datasets that are "interesting" but still in 2D. The main observations to make are: single linkage is ...

https://scikit-learn.org/stable/auto_examples/cluster/plot_linkage_comparison.html



We can look at this to see what kind of data is good for which type of Methods

1. Single Linkage

- Takes the intergroup dissimilarity to be that of the **closest** (least dissimilar) pair

$$d_S(G, H) = \min_{i \in G, j \in H} d(x_i, x_j)$$

2. Complete Linkage

- Takes the intergroup dissimilarity to be that of the **furthest** (most dissimilar) pair

$$d_C(G, H) = \max_{i \in G, j \in H} d(x_i, x_j)$$

3. Average Linkage

- Utilizes the **average** of all pairwise dissimilarities between the groups

$$d_A(G, H) = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{j \in H} d(x_i, x_j)$$

4. Ward Linkage

- Uses a more complicated distance to **minimise the variance** within groups. It usually returns more compact clusters than the others. Suppose that group G was formed by merging groups G1 and G2.

$$d_W(G, H) = \sqrt{\frac{|H| + |G_1|}{N_G + N_H} d_W(H, G_1)^2 + \frac{|H| + |G_2|}{N_G + N_H} d_W(H, G_2)^2 + \frac{H}{N_G + N_H} d_W(G_1, G_2)^2}$$

▼ Steps

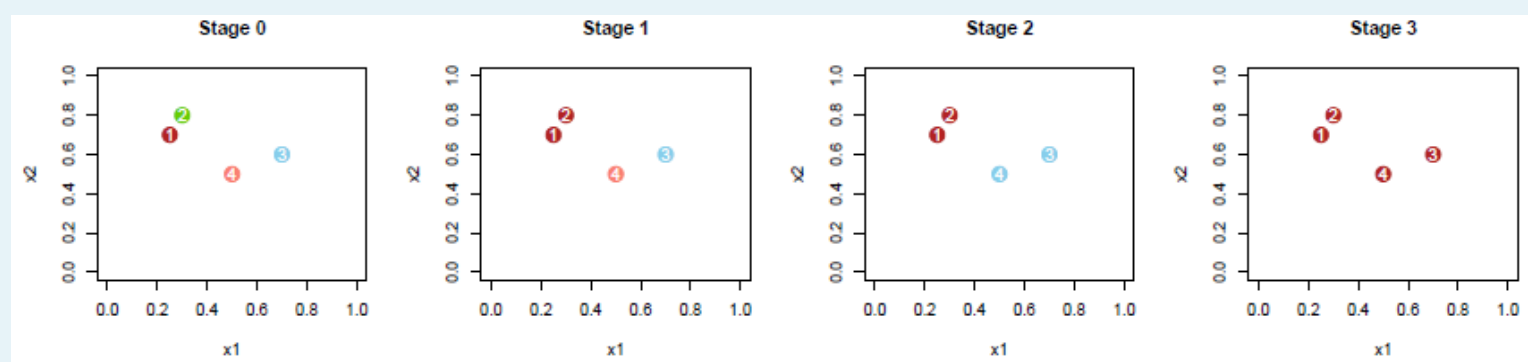
Note that each stage, for which groups to combine, it also depends on the type of linkage method that we choose to use. We will compute "pairwise" dissimilarities for each stage using the type of linkage method and choose the shortest one to combine

1. We will start with N clusters first where all the points are 1 cluster each
2. We will compute the pairwise distances between each of the points (using any of the dissimilarity measure that we had for those between individual measures)

↓ Point →	1	2	3	4
1	0.00	0.11	0.46	0.32
2	0.11	0.00	0.45	0.36
3	0.46	0.45	0.00	0.22
4	0.32	0.36	0.22	0.00

Example of the pairwise distances

3. If there are groups already formed, we will use one of the linkage methods to find the dissimilarity with other groups. We will try to find the "pairwise" distance between points and groups and we will choose the ones with the smallest distance and combine them together
4. For every stage, we will reduce the number of clusters until we are left with 1 cluster (which we would have taken $K - 1$ stages to complete)



Example of the combination of clusters at each stage

▼ Properties that we want to see

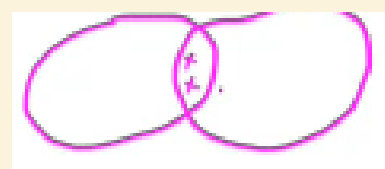
• Compactness

- Points within a cluster should be very close to each other
- Violated by **Single Linkage**
 - Our groups will look very long (big in diameter) as data that are very far apart will be clustered together because we are looking at the shortest distance between the groups



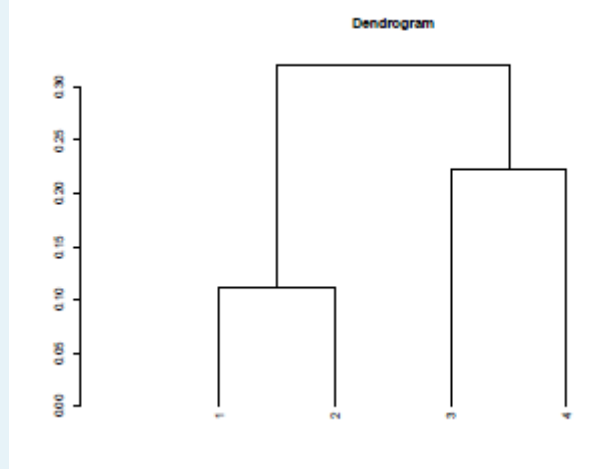
• Closeness

- Points should only be close to those points that are in their cluster and not to other cluster (Far from other clusters)
- Violated by **Complete Linkage**
 - Clusters will be very close to each other and the clustering of some points could be dubious



▼ Visualisation of the Hierarchy of Clusters

• Dendrograms

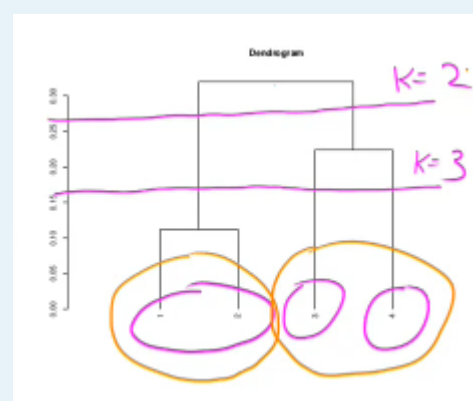


Example of Dendrogram

- Leaves are the points
- The axis on the left allows us to see at which distance did the points / groups combine. We can read off the distance by drawing a horizontal line at the point of intersection between 2 lines and that will be the distance
- Height of each node is proportional to the value of the intergroup dissimilarity between its 2 child nodes
- We can see that the group on the left is less dissimilar to the group on the right since the distance which they combine is smaller

▼ Choosing Clusters

- We can make a cut at the intersection points to get the different numbering of clusters, we can see how many single lines there are at the points that we cut



Example of how we cut to form the clusters

▼ Choosing of Number of Clusters

▼ Silhouette Measure

- Allows us to measure how well the clustering result is and we can decide the number of clusters that we should use
- To get the "score" for how well a particular clustering is, we can take the average of all the silhouette values and compare across different number of clusters

▼ Formula

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max(b(x_i), a(x_i))}$$

$b(x_i)$ - Distance from x_i to the nearest cluster that x_i **DOES NOT** belong to (Closeness Property) (We want this value to be **BIG**)

$a(x_i)$ - Average distance to other points that are in the **SAME** cluster as x_i . (Compactness Property) (We want this value to be **SMALL**)

▼ Range

Our values can range from any of the numbers between the range

💡 $-1 \leq s(x_i) \leq 1$

- 1 - Means that the clustering result is **good**
- -1 - Means that the clustering result is **bad**. The point could be better if it is in another cluster

▼ Functions in R

▼ Preparation

- **Correlation Matrix** - Use `cor()` and we pass in the data frame with the variables we want in the columns

▼ Conversion to Distance Matrix from Correlation Matrix

- $\frac{1-\rho}{2}$ - We will make use of this to change the correlation matrix to a distance matrix
- We need to make this conversion because distance is more than 0 and correlation coefficient ranges to -1
- Note that:
 - $-1 \leq \rho \leq 1$ - This is the range for correlation coefficient
 - 1 - Most **Similar**
 - -1 - Most **Dissimilar**
 - $0 \leq \frac{1-\rho}{2} \leq 1$ - This is the range for the distance matrix

Note that we can put in the values from the correlation coefficient to check

- 0 - Most **Similar**
- 1 - Most **Dissimilar**

- **Distance Object** - Use `as.dist()` and we pass in the converted correlation matrix using the formula into this to create a distance object.
 - It removes the diagonals and the upper triangle since the matrix is symmetric and it will be a more efficient way of storing the data

▼ Hierarchical Clustering

`hclust()` - Function that will compute the hierarchical clustering

- `d` - We will need to pass in a distance object which we have prepared into this which is our dissimilarity matrix
- `method` - The method that we are using for our linkage
 - `"single"`
 - `"complete"`
 - `"average"`
 - `"ward.D"` - When our distance object is a squared value
 - `"ward.D2"` - When our distance object is on a scale of the distances (without squaring it)

`cutree()` - Helps to cut up the tree into groups

`tree` - `hclust` object that we have created

`k` - Number of cluster that we want to cut into

▼ Dendrogram

Note that we can plot the dendrogram object to view the structure of it after we have converted it to a dendrogram

`as.dendrogram()` - Creates a dendrogram object for our hierarchical clustering object

- `object` - We can just pass in the hierarchical clustering object that is created from `hclust()`

`ord.dendrogram()` - Orders the leaves of a dendrogram

- `x` - Dendrogram object
- This ordering is from the last to join the cluster to the first that is clustered together

▼ Silhouette Measure

`library(cluster)` - We will need to load this library before using

`silhouette()` - Computes the Silhouette measure for all the observations

- `x` - Ordering for which observations goes to which cluster. Can be taken from `cutree()`
- `dist` - Distance matrix that we have from
- Note that we can apply `mean` to the third column to get the mean score for this clustering
- We can make use of `sapply()` to loop through a number of sizes for the clusters and the one with the highest mean score is the best

```
grpings <- lapply(2:7, cutree, tree=hc)
sapply(grpings, function(x) mean(silhouette(x, as.dist((1 - cor_Cars93)/2))[,3]) )
```

▼ Dissimilarity Measure Computation

`daisy()` - Helps to compute the pairwise dissimilarities between observations

`x` - Data frame or vector that we want to compute the pairwise dissimilarities for

`metric` - Specify what is the metric that we are using to compute dissimilarities between observations

- `euclidean` - Euclidean distance (default)
- `manhattan` - Manhattan distance
- `gower` - Gower's Distance (Can be used when we have both categorical and numerical data)

▼ Example Codes

```
hc <- hclust(as.dist((1 - cor_Cars93)/2), method = "ward.D2")
ord <- order.dendrogram(as.dendrogram(hc))
# plotting the structure of the dendrogram
plot(as.dendrogram(hc))
# ordering the data using the order for the dendrogram
cor_Cars93_df2 <- mutate(cor_Cars93_df,
  var1 = factor(var1,
    levels=row.names(cor_Cars93)[ord]),
  var2 = factor(var2,
    levels=row.names(cor_Cars93)[ord]))
ggplot(cor_Cars93_df2) +
  geom_tile(aes(x=var1, y=var2, fill=correlation)) +
  scale_fill_gradient2() +
  theme(axis.text.x=element_text(angle=90, vjust=0,
    hjust=1))

# silhouette code
library(cluster)
grp_3 <- cutree(hc, k=3)
sil_out <- silhouette(grp_3, as.dist((1 - cor_Cars93)/2))

# get groupings for k= 2 to 7
grpings <- lapply(2:7, cutree, tree=hc)
sapply(grpings, function(x) mean(silhouette(x, as.dist((1 - cor_Cars93)/2))[,3]) )

# dissimilarities between cars:
Cars93 %>% select(-1, -2, -27) %>%
  daisy(metric="gower") -> d2
```

▼ Multidimensional Scaling

▼ Definition

- Plot high dimensional data on 2-D Plane
- With a choice of k, we seek values $z_1, z_2, \dots, z_N \in \mathbb{R}^k$ such that this function is minimised:

$$S(z_1, \dots, z_N) = \left[\sum_{i \neq j} (d(x_i, x_j) - \|z_i - z_j\|)^2 \right]^{1/2}$$

▼ Alternatives

- Principle Component Analysis (PCA)
- Projection Pursuit
- Bi Plots
- Cosine Similarity

▼ Note

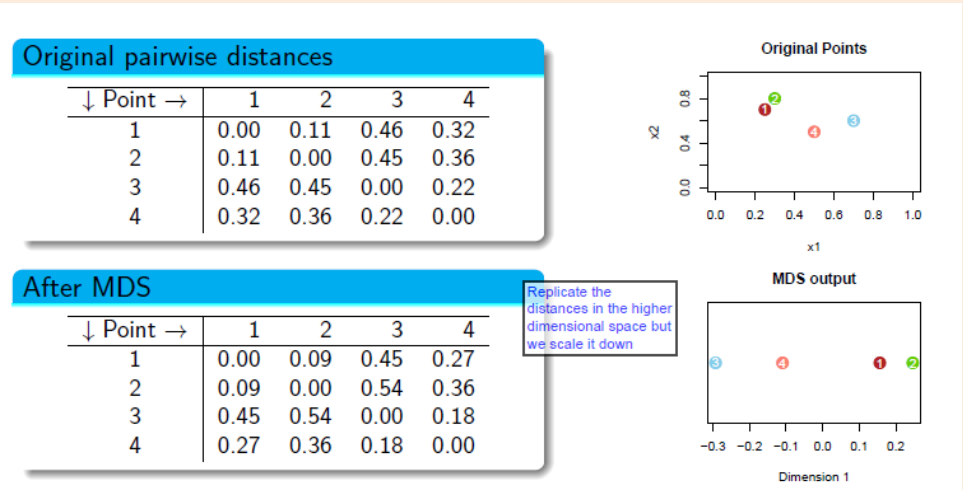
- MDS is **not** the same as Principal Component Analysis (PCA):
 - PCA maximises **variance**, orthogonal to earlier components.
 - Principal components are ordered; MDS are not.
 - Principal components are linear combinations of the original vectors; MDS output is not.
- Due to the scaling down, there will be loss of information and we should not just read off the exact distances between points from this as it is not exact.

▼ Usage

- Normally:
 1. Use Hierarchical Clustering to find the optimal clusters
 2. Plot using MDS so the variables are on a similar scale
 3. Color the points using the clusters that are computed from Hierarchical Clustering

▼ Steps

1. We will still need to compute the pairwise distances between the points like what we did for Hierarchical Clustering
2. We will then find some sort of coordinates in only 1/2 dimension that still maintains the pairwise distances between the points
 - a. Note that the original dataset could have many dimensions and we will want to minimise it down to 1/2 dimension so that we can plot it



Example

▼ Metric MDS

- Note that if it is a metric MDS, it tries to maintain the distances explicitly as far as possible
- Non-metric MDS, only preserves the relative ordering

▼ Functions

`cmdscale()` - Metric MDS which computes the coordinates that it can be scaled down to

- `d` - Distance object that we have for the pairwise dissimilarities
- `k` - The maximum dimension of the space which the data are to be represented in

`MASS::sammon()` - Non Metric MDS which computes the coordinates that it can be scaled down to

- `d` - Distance object that we have for the pairwise dissimilarities
- `k` - The maximum dimension of the space which the data are to be represented in

Transformation of Data

▼ Usage

- When a relationship does not seem linear, we will want to try to transform it so that it is linear and becomes easier for us to visualise it

▼ Ladder of Transformation

- Going Rightwards along the ladder makes the values larger and can help to correct for negative skew
 - Makes the larger numbers bigger without expanding the smaller numbers too much
- Going Leftwards along the ladder makes the values smaller and can help to correct for positive skew
 - Makes the larger numbers smaller without squashing the smaller numbers too much

▼ Note

- We can think of the direction in which we may want to expand or make it smaller
- Note that we may need to **take powers in between** the transformations and we will need to try to see which are the transformations that could best make it a linear relationship
- Note that when we take **negative powers**, we need to put the negative sign so that we can keep the relative relation between the points as taking negative powers will make the larger values smaller than the smaller values
- Note that if we use the `scale` functions in ggplot, it will make the scale not be of equal distance anymore and it will be the representative value of the scales that we use
 - E.g log scale. If we have the value 100 on the x axis and we use the log scale, it will not change the x axis to 1, 2, 3. It will be $10^1, 10^2, \dots$ (**Note that the distances are not the same**)

▼ For y Variable

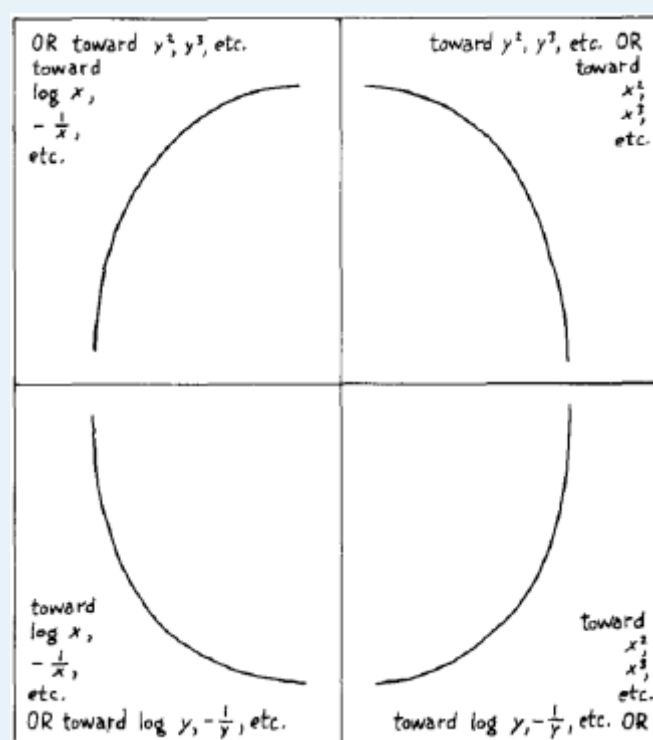
$$-y^{-2}, -y^{-1}, \log y, y, y^2, y^3$$

- Going leftwards makes the values smaller
- Going rightwards makes the values larger

▼ For x Variable

$$-x^{-2}, -x^{-1}, \log x, x, x^2, x^3$$

- Going leftwards makes the values smaller
- Going rightwards makes the values larger



We can use this diagram to guide us what transformations to use

▼ Other Ways to find Optimal Transformation

- `boxcox`

- Optimises the coefficient of variation
- `yeojohnson`

Other Tables

▼ Bivariate Plots

- Made using `ggbivariate()` under `GGally()`
- `ggbivariate()`
 - `outcome` - Outcome variable that we want to test against
 - `explanatory` - Vector of explanatory variables that we want to test against the outcome variable and see what is the association
 - This shows us how each of the explanatory variables affect the outcome variables

▼ GGPairs Plots

- `ggpairs()` under `GGally`
- Can be used when we want to plot many different variables against each other and it will decide the kind of plots that it should make for each combination of variables and we can modify it as well
- `ggpairs()`
 - `data` - Data frame that contains the data
 - `aes()` - Aesthetic mappings other than the x and y since x and y is going to be all combination of columns
 - `columns` - Which columns to use. Default is to use all
 - Picking the plots:
 - `lower` - We can specify the kind of plots that we want for the lower triangle (for continuous, discrete or combo)
 - `upper` - We can specify the kind of plots that we want for the upper triangle (for continuous, discrete or combo)
 - `diag` - We can specify the kind of plots that we want for the diagonals (for continuous, discrete or combo)
 - `vig_ggally("ggally_plots")` - We can check this for the possible plots
 - We can supply the `NAME` under `ggally_NAME` (These are the various plot functions)
 - We can use `wrap("FUNC_NAME", func_args)` - Use the wrap function to put in the arguments for those plot functions as well
 - `labeller` - Choose the labels for the plot
 - `as.labeller(c(`variable_name` = "new_label", ...))`
 - `showStrips`

Additional Statistical Knowledge

▼ Stochastic Ordering

$$X \geq Y \\ \Leftrightarrow F_x(z) \leq F_y(z) \quad \forall z \in \mathbb{R}$$

- X is **stochastically greater** than Y if the CDF of X is lower than Y for all values of z
- We can think of it as since the cumulative distribution for X is always lower, it means that there are lesser values of X that have lower values, therefore, it will mean that the values of X is expected to be larger than Y

$$\mathbf{X \text{ and } Y \text{ independent}} \\ \Rightarrow P(X > Y) \geq P(Y > X)$$

- This useful when we are analysing cumulative distribution plots. If there are cumulative distributions that have the CDF of one lower than the other for all values, then we can say that we can expected the values of X to be larger than the value of Y if we take any random sample

▼ Chi-Squared Test for Independence

- Measure of the strength of the difference between observed and expected values

▼ Test Statistics

$$\chi^2 = \sum_{i,j=1}^n \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

- Note that we will sum up all the standard residuals

▼ Steps

1. Under the assumption of independence, we will compute the expected counts for each of the values
2. We note that the joint probability for each of the cells will be the multiplication of each of the marginal probability since they are independent

Breast Cancer	PMH user	PMH non user	
Absent	121	682	803
Present	79	318	397
	200	1000	1200

3. Compute expected values of each of the cells based on the assumption of independence.
 - a. For example, PMH User with Breast Cancer:
Expected Count = $n \times P(PMH, Present)$
 $= n \times P(PMH) \times P(Present)$
 $= 1200 \times \frac{200}{1200} \times \frac{397}{1200}$
 ≈ 66.167
4. Compute the residual values by taking **Observed - Expected**
5. If we want to compute the standardised residuals we take **$(Observed - Expected^2)/Expected$**
 - a. This is useful because some of the values could be abnormally large and it could cause the analysis to be inaccurate
 - b. Guidelines:
 - i. ≤ -2 , normally means that the observed frequency is lesser than the expected value a little
 - ii. ≥ 2 , normally means that the observed frequency is larger than the expected value by a little
 - iii. ± 3 means that something unusual is happening