Visualisation

Exploring data through visualisation

Vik Gopal

Although we often hear that data speak for themselves,
their voices can be soft and sly.

Introduction

- In this chapter, we shall put the techniques and principles that we have covered so far into practice.
- We shall try to suggest which plots should be used in particular situations, and techniques of plotting.
- We shall also highlight what to look out for and discuss in these plots.

The Process of Making Plots

- In this chapter, we focus on visualisation.
- Making a graph or a plot should not be a matter of dumping your data into a software and then pasting the plot into PowerPoint.
- As John W. Tukey said,

  There is no excuse for failing to plot and look.
- He also said

  The greatest value of a picture is when it *forces* us to notice **what we never expected to see**.

Exploratory Data Analysis (EDA)

- Visualisation is an integral part of EDA.
- It is a highly iterative process. We should expect to:
  1. Generate questions about our data.
  2. Search for answers by visualising, transforming and modeling our data.
  3. Use what we learn to refine our questions and/or generate new questions.
- The final plot we submit or show should be carefully designed.
  - 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.

Questions

When we inspect our data, the questions that we generate almost always fall into the umbrella categories:

1. What type of variation occurs within my variables?
2. What type of covariation occurs between my variables?

Making Comparisons

- Once again, a quote from Tukey:

  Two kinds of comparisons come up in the simplest of common language:

  "*Bill is a head taller than Jim.*"

  "*George weighs twice as much as his brother Jack.*"

- Both these kinds of comparisons are useful. We should try to add text/explanations that interpret the differences in our charts in these terms.
- The first kind, based on addition, is simpler.
- However, there are times, when we have to use the second kind of comparison, which is based on multiplication.
- Logarithms were invented to convert multiplication to addition.

Definitions

Let's recall some terms we first encountered when tidying our data:

- 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.

General Guidelines

what we should discuss

When we display a chart, here are a few general guidelines regarding what to discuss:

1. Which values are the most common? Can we say why?
2. Which values are rare? Why?
3. Are there any unusual patterns in the data?

Diamond Quality Bar Chart

example

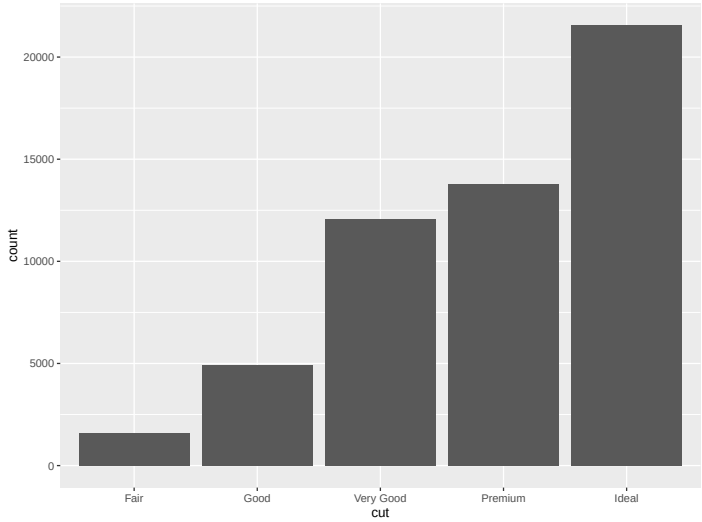
- We shall work with the `diamonds` dataset that comes with the `tidyverse`.
- Here, we shall create a bar chart of the `cut` variable. It is an ordered factor with the following levels:

  `Levels: Fair < Good < Very Good < Premium < Ideal`

```
library(tidyverse)
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut))
```

Diamond Quality Bar Chart

plot



Diamond Quality Bar Chart

what we can say

1. The modal category is *Ideal*. This is the highest quality cut possible.
2. The count for *Ideal* is approximately 30% higher than the next most frequent category.
3. The category with the lowest count is *Fair*.
4. The counts for *Very Good* and *Premium* are close to each other.

Contingency Tables

- A *contingency table* is a display for two categorical variables.
- Its rows list the categories for one variable and its columns list the categories of the other variable.
- Each entry in the table is the number of observations in the sample at a particular combination of categories of the two variables.

2x2 Contingency Table

- The simplest contingency table is one that has 2 rows and 2 columns.
- Here is an example from a breast cancer study.

Breast Cancer	PMH user	PMH non user
Absent	121	682
Present	79	318

2x2 Contingency Table

cont'd

- It appears that the rate of breast cancer is higher for PMH users than non-users. (39.5% versus 31.8%)
- This is precisely what we mean by "association". A higher proportion of breast cancer is *associated* with PMH users.
- Two categorical variables are *independent* if the conditional proportions for one of them are identical at each category of the other.
- The variables are *dependent*, or associated, if the conditional proportions are not identical.

Describing the Difference

using difference

We let $\hat{p}_1$ and $\hat{p}_2$ be the proportions from the two groups. For instance, in the PMH user example, $\hat{p}_1 = 0.318$ and $\hat{p}_2 = 0.395$.

- The difference of proportion is defined to be:

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

- Thus we can say that the difference in proportions is 0.077, or 7.7%.
- Note that the difference in proportions can take any value between -1 and 1.

- A value of 0 corresponds to "no association".
- It does not matter whether we take $\hat{p}_1 - \hat{p}_2$ or $\hat{p}_2 - \hat{p}_1$. The strength of the association does not change; only the sign of this metric does.

Describing the Difference

using relative risk

- The relative risk is defined to be

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

- The relative risk can take on values between 0 and infinity. A value of 1 corresponds to "no association".
- Analogous to the difference in proportions, it does not matter whether we take $\hat{p}_1/\hat{p}_2$ or $\hat{p}_2/\hat{p}_1$. A relative risk of 0.25 is the same as a relative risk of 4.0.
- It is preferrable to use relative risk when both proportions are close to 0 or 1.

0. For instance, suppose that in the PMH users example, we had observed that $\hat{p}_1 = 0.01$ and $\hat{p}_2 = 0.03$. Which statement below conveys more insight?
   - The proportion of breast cancer incidence for PMH users is 3 times higher than that for PMH non-users.
   - The difference in proportions between PMH users and non-users is 0.02 or 2%.

Definition of Odds

- For a categorical variable with 2 possible values, define one of them to be the *success* and the other to be the *failure*.
- Let p be the probability of success, and $1 - p$ be the probability of failure.
- Then the **odds of success** is defined to be

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

- 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.

Example of Odds in Football

- Just before the 2014 FIFA World Cup began, the bookmakers listed the odds of Brazil winning the trophy as 3/1.
- This just means the odds were 3.
- When we convert it to a probability, what they are saying is that the probability of Brazil winning the World Cup was 0.25.
- This can be obtained by solving for p in the equation

$$\begin{aligned} 3 &= \frac{p}{1-p} \\ 3-3p &= p \\ 3 &= 4p \\ \frac{3}{4} &= p \end{aligned}$$

An Aside on Logarithms

- The logarithm of a value x is defined to by y such that

$$x = 10^y$$

  We write it as

$$y = \log_{10} x$$

- In computations, the most important property of logs is that

$$\log_{10}(x_1 x_2) = \log_{10} x_1 + \log_{10} x_2$$

- Thus the log of a ratio is the difference of the logs of numerator and denominator.

$$\log_{10}(x_1/x_2) = \log_{10} x_1 - \log_{10} x_2$$

  This property enables us to deal with subtraction instead of division.

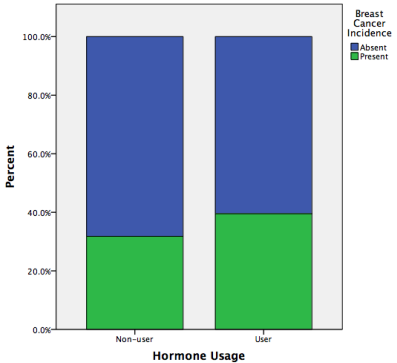
- Another important property of logs is that it de-emphasises large differences. This is best understood from the graph; it enables us to make a distribution of values more symmetric.

Describing the Difference

using odds ratios

- In statistics, the **odds ratio** - the ratio between the odds of success between two groups is used quite often because, due to the sampling design, it is not possible to estimate the relative risk.
- In our case, we consider taking a transformation of the odds and then comparing the transformed odds.
- The transformation is the log-transformation. It accentuates differences when the proportions for both groups are close to 0 or 1. For instance,
  - Suppose $\hat{p}_1 = 0.99$ and $\hat{p}_2 = 0.97$. Then the log odds for group 1 is 1.996 and the log odds for group 2 is 1.590.
- On the log-odds 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 $+\infty$ corresponds to probability of success equal to 1.

A Stacked Bar Plot



- We can visualise a contingency table using a stacked bar chart like the one on the left.
- Let's think about how we can visualise contingency tables with more than 2 rows and/or columns.

Contingency Table for Cut and Colour

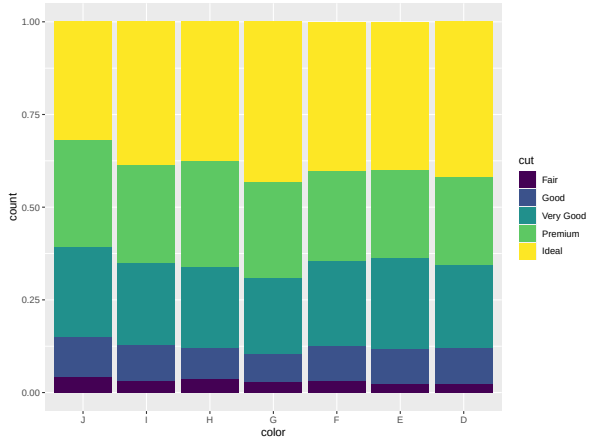
- We have to reorder the levels for color so that D is the best (greatest) and J is the worst (least).

```
d2 <- mutate(diamonds,
             color = factor(color, levels=rev(levels(color)),
             ordered = TRUE))
d2 %>% count(color, cut) %>%
  pivot_wider(names_from = "color", values_from="n") %>%
  arrange(desc(cut))
```

Cut/Color	J	I	H	G	F	E	D
Ideal	896	2093	3115	4884	3826	3903	2834
Premium	808	1428	2360	2924	2331	2337	1603
Very Good	678	1204	1824	2299	2164	2400	1513
Good	307	522	702	871	909	933	662
Fair	119	175	303	314	312	224	163

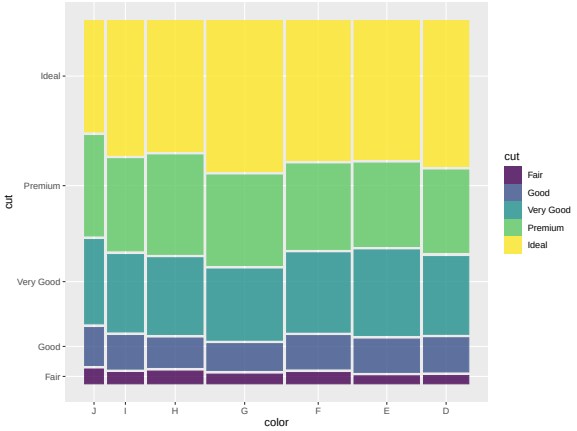
Stacked Bar Chart in ggplot

```
library(ggmosaic)
ggplot(d2)+ geom_bar(aes(x=color, fill=cut),
                     position=position_fill(reverse=TRUE))
```



Mosaic Charts in ggplot

- Mosaic charts are similar to stacked bar charts, but the size of the rectangles represent the counts instead of proportions.
- It is easier to tell marginal counts with these.



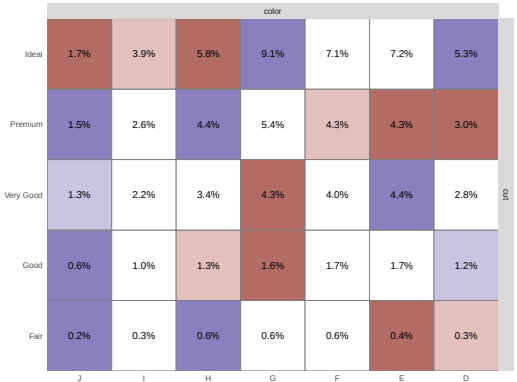

```
ggplot(d2) +
  geom_mosaic(aes(x=product(color), fill=cut))
```

Plotting Tables

- It is of course possible to plot the actual table, displaying proportions as well.

```
library(GGally)
ggtable(d2, "color", "cut",
        cells="prop",
        fill="std.res")
```

- The blue and red shades identify which cells have higher and/or lower counts than expected *under the assumption of independence* between the two variables.



Computing Proportions Under Independence

```
cont_table <- d2 %>% count(color, cut) %>%
  pivot_wider(names_from = "color", values_from="n") %>%
  arrange(desc(cut))
col_tmp <- colSums(cont_table[,-1])
row_tmp <- rowSums(cont_table[,-1])
exp_props <- (row_tmp/sum(row_tmp)) %*% t(col_tmp/sum(col_tmp)) *100
exp_props <- data.frame(exp_props) %>%
  mutate(cut = cont_table$cut, .before=J)
```

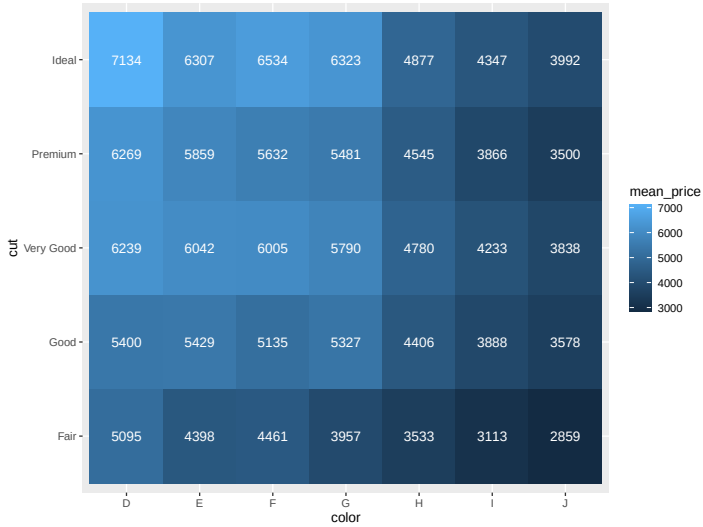
Pivot Tables

- Thus far, we have dealt with tables of counts.
- Sometimes, we have a third variable within the cell, which is not count.
- Consider studying diamonds that are approximately 1 carat in weight.

```
price_table <- diamonds %>% filter(between(carat, 0.95, 1.05)) %>%
  group_by(cut, color) %>%
  summarise(mean_price =mean(price), .groups="drop") %>%
  mutate(text_col=if_else(mean_price <= 4900, "white", "black"))

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)),
            colour = "white")
```

Pivot Tables

plot



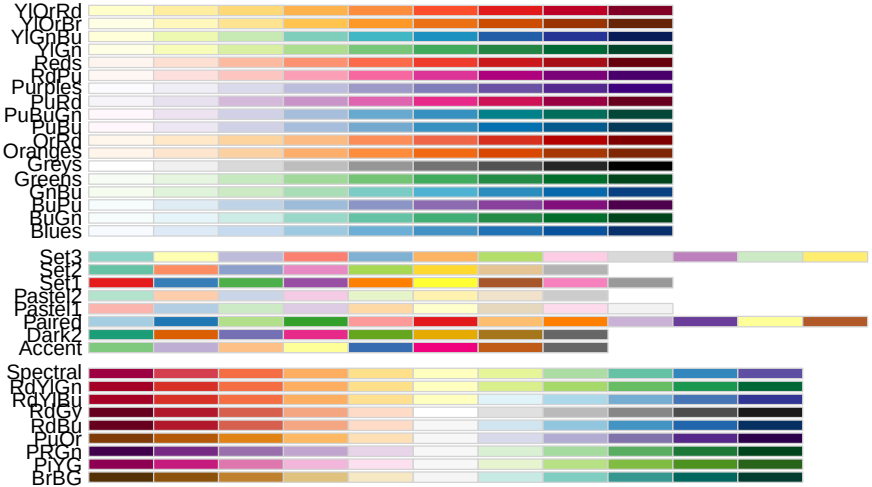
Colour Palettes

- A colour palette is a range of colours.
- Typically these can be used in plots to distinguish between groups of data, e.g. males and females, ethnicity, etc.
- Colours can be used for continuous data as well, by binning the range of continuous values and using a different colour for each bin.
- The functions `scale_fill_` and `scale_colour_`
  - can take a colour palette as an input, or
  - create a colour palette based on specified "end" colours.

Representation of Colours in R

- Within R, each colour is represented as a sequence of 8 hexademical digits.
  - The first (leftmost) 6 digits indicate the colour in the RGB palette.
  - The final (rightmost) 2 digits indicate the transparency of the colour displayed.
- The `RColorBrewer` package provides a set of palettes that are widely used. They are tightly integrated into `ggplot2`.

RColorBrewer Palettes



Which Palette To Use?

- Colour can be thought of as a three-dimensional concept consisting of
  - **hue**: e.g. red, green or blue.
  - **value**: light versus dark.
  - **saturation**: e.g. dull versus vivid.
- Brewer notes that viewers tend to perceive *differences* between colours most readily when changing hue, and perceive *ordering* most readily when lightness (value) is changed.
- For instance, a map viewer can quickly tell that a green region is somehow different from a red region, but can more readily report that the light green region is "lower" than the dark green region.

Sequential Versus Diverging Patterns

- When we want our reader to identify which values which are higher or lower than other values, then we should use sequential values.
- This is because lightness is best suited for sequential perception.
- If we would like our viewer to identify ordering in two directions, then we should use a diverging pattern, with different hues.
- For instance, if we assign blue to regions having lower than average rates and red to regions having higher than average rates, then the reader can quickly separate the blue from the red regions.
- If we vary lightness with each hue, then we readers can also order the regions with the two divergent directions.

Using a Heat Map

- A common task is to understand a correlation matrix, or in general, a matrix that represents pair-wise distances between objects.
- A first approach uses a heat-map to represent correlations. In technical terms, the correlation is mapped to the fill aesthetic.
- From the MASS package, the Cars93 dataset contains several variables, measured on cars.

```
data(Cars93, package = "MASS")
cor_Cars93 <- select(Cars93,
                     which(!sapply(Cars93,
                                   is.factor))) %>%
  cor(. , use="pair")

cor_Cars93_df <- as.data.frame(cor_Cars93,
                               row.names=NULL ) %>%
  mutate(var1 = row.names(cor_Cars93)) %>%
  pivot_longer(Min.Price:Weight, names_to="var2",
               values_to="correlation")
```

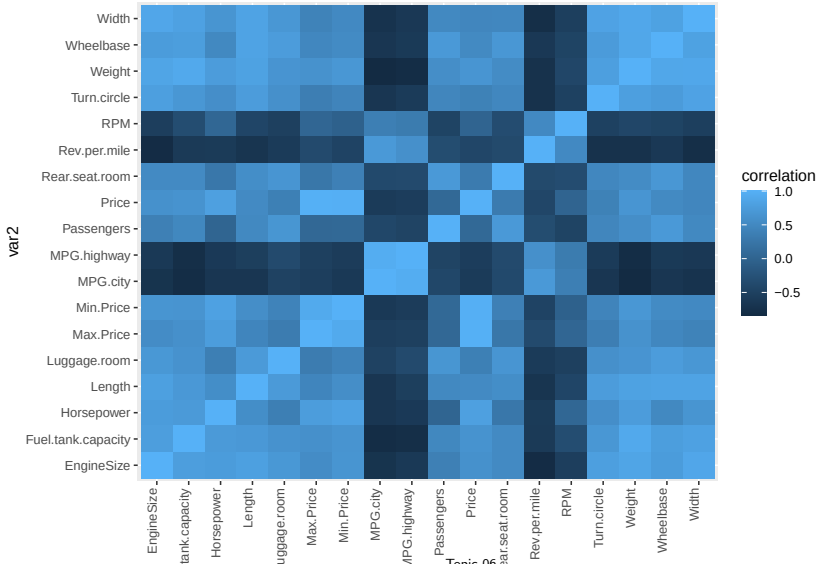
Representing Correlations

- The resultant plot can be improved.
- The choice of colours could be more judicious.
- The row and columns could be ordered more intelligently.

```
ggplot(cor_Cars93_df) +
  geom_tile(aes(x=var1, y=var2, fill=correlation)) +
  theme(axis.text.x=element_text(angle=90,
                                 vjust=0, hjust=1))
```

Representing Correlations

a heat map



Re-ordering Rows and Columns

a more intelligent presentation

- The variables do not have any inherent ordering, and so we can try to order them to place similar correlations near to one another.
- For that, we have to perform a clustering of the numbers first.
- This is similar to what we did for the interval generation for the purpose of choosing colours.

About Hierarchical Clustering

- Hierarchical clustering methods produce a hierarchical representation of clusters in a dataset.
- Unlike alternative methods, such as K-means, they do not require a prior specification of the number of clusters.
- If your data contains N observations, the largest number of possible clusters is N, and the least is 1.
- At each possible cluster size, there is an optimal division of your data, for the algorithm and criterion that we choose.
- The output of a hierarchical clustering allows you to visualise the optimal $N - 1$ possible clusterings at one go, using a **dendrogram**.

Hierarchical Clustering Input

- Cluster analysis is an attempt to identify groups within the data, such that members within a group are "similar" to one another.
- In order to proceed, we need to formalise this idea of similarity/dissimilarity.
- Suppose that we have N observations $x_1, x_2, \ldots, x_N$ and we wish to group them into K clusters. Each observation is typically a vector of p measurements.
- We write $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ for instance.
- The Cars93 example had 93 rows and 18 columns. However, **we are going to cluster the variables**, so for our purpose, $N = 18$ and $p = 93$.

Dissimilarity Measures

Between individual observations

- Let d be a binary function of two observations, that measures pairwise dissimilarity.
- One of the most common choices is Euclidean distance:

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

- Another common choice is the $L1$-norm:

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

Between groups

- We build on this pairwise dissimilarity to obtain a measure of dissimilarity between groups.
- Suppose we have two groups of points G and H, with N_G and N_H points within them respectively.
- We decide upon a **linkage method** to compute the dissimilarity between groups, using only point-wise dissimilarities.

Linkage Methods

- Single linkage takes the intergroup dissimilarity to be that of the closest (least dissimilar) pair.

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

- Complete linkage takes the intergroup dissimilarity to be that of the furthest (most dissimilar) pair.

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

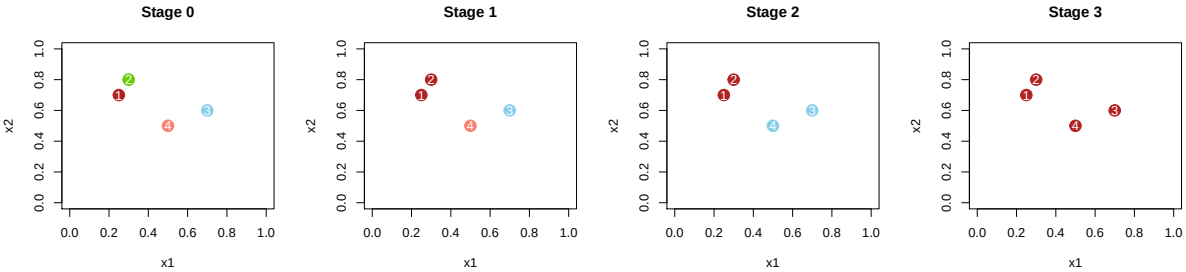
- Average linkage utilises the average of all pairwise dissimilarities between the groups:

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

- 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 G_1 and G_2. Then the Ward distance between groups is

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

Algorithm Walk-through



Data points

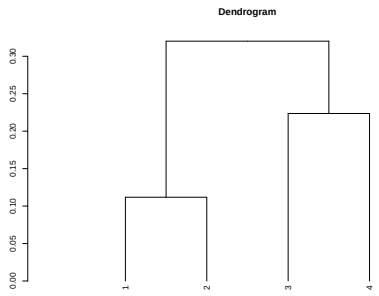
Point	x1	x2
1	0.25	0.70
2	0.30	0.80
3	0.70	0.60
4	0.50	0.50

Pairwise distances

↓ 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

Visualising the Hierarchy of Clusters

- The dendrogram shows that points with indices 1 and 2 (the closest two points) merge at a small vertical distance, but the points 3 and 4 merge at a much higher vertical distance.
- This shows that points 1 and 2 are less dissimilar to one another than points 3 and 4.
- In other words, the height of each node is proportional to the value of the intergroup dissimilarity between its two child nodes.



Back to Cars93 Dataset

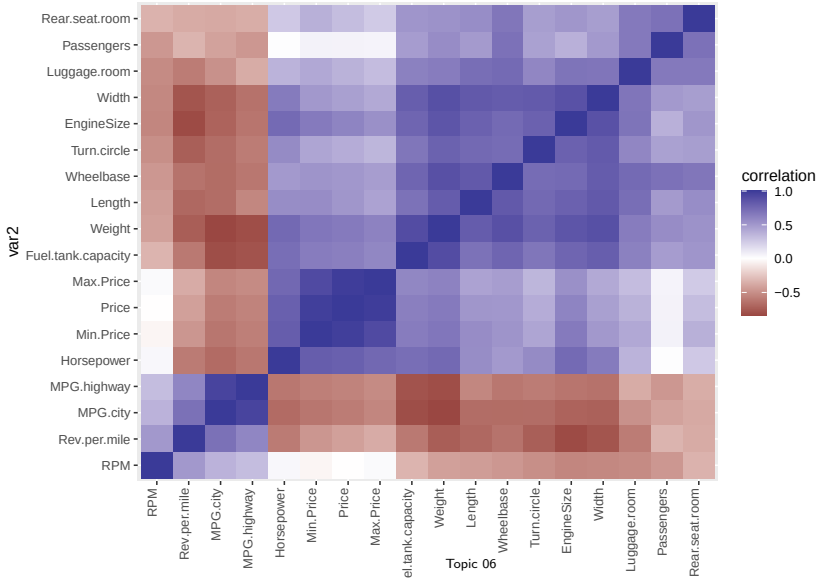
code

- Earlier, we computed the pairwise correlations between variables in the Cars93 dataset.
- Let us now use that matrix as the distance matrix input to the clustering algorithm.

```
hc <-hclust(as.dist((1 - cor_Cars93)/2))
ord <- order.dendrogram(as.dendrogram(hc))
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))
```

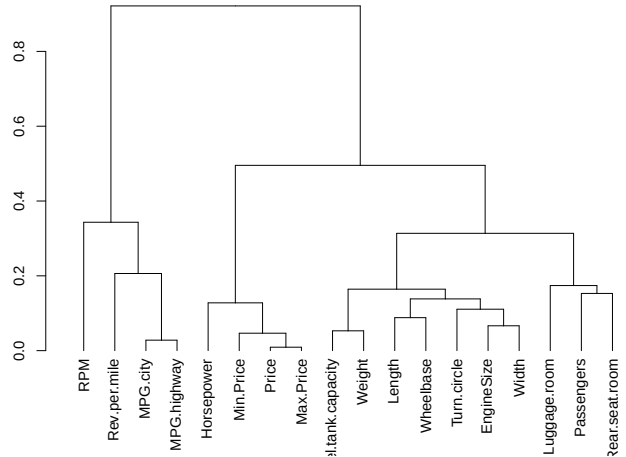
Back to Cars93 Dataset

ordered correlation matrix



Back to Cars93 Dataset

the dendrogram



- The dendrogram itself provides information on which variables are similar to one another.

About Multi-Dimensional Scaling

- An alternative approach to visualising similarities of high-dimensional data is provided by Multi-Dimensional Scaling.
- Once again, suppose we have computed all pairwise dissimilarities $d(x_i, x_j)$ between our N high-dimensional vectors.
- With a choice of k, we seek values $z_1, z_2, \ldots, z_N \in \mathbb{R}^k$ such that the following function is minimised:

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

Warning

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.

Preserving Pairwise Distances

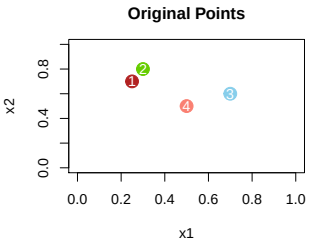
Let us return to our earlier example, from page 47.

Original pairwise distances

↓ 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

After MDS

↓ Point →	1	2	3	4
1	0.00	0.09	0.45	0.27
2	0.09	0.00	0.54	0.36
3	0.45	0.54	0.00	0.18
4	0.27	0.36	0.18	0.00

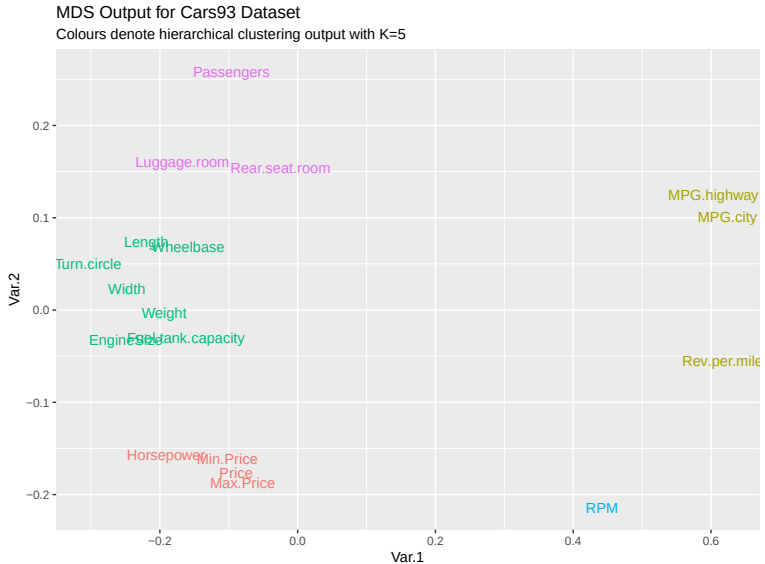


Multi-Dimensional Scaling

code

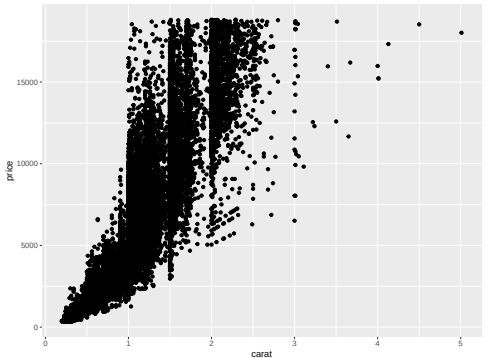
```
cars_dist <- as.dist((1 - cor_Cars93)/2)
mds2 <- MASS::sammon(cars_dist, k = 2)
grps <- as.factor(cutree(hc, k=5))
mds_df <- data.frame(mds2$points) %>%
  mutate(label = row.names(mds2$points), Cluster=grps) %>%
  rename('Var.1' = 'X1', 'Var.2'='X2')

ggplot(mds_df) +
  geom_text(aes(x=Var.1, y=Var.2, label=label, col=Cluster),
            show.legend = FALSE) +
  labs(title="MDS Output for Cars93 Dataset",
  subtitle = "Colours denote hierarchical clustering output with K=5")
```

Multi-Dimensional Scaling



Two Continuous Variables

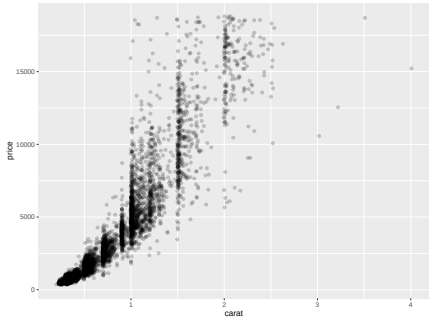
- Scatterplots, or the point geom, are used to visualise the relationship between two continuous variables.
- Consider the variables `carat` and price from the `diamonds` dataset.




```
ggplot(diamonds) +
  geom_point(aes(x=carat, y=price))
```

Too Many Points?

- Earlier, we saw that one solution to overplotting was transparency.
- Another solution is to sample a set of points and then plot them.




```
set.seed(11)
sub_diamonds <- sample_n(diamonds, size=5000)
ggplot(sub_diamonds) +
  geom_point(aes(x=carat, y=price), alpha=0.2,
             position="jitter")
```

Transformation of Data

- If a relationship does not appear to be linear, it is our job as analysts to find a succint relationship for the data.
- There are several transformations we can make in order to make the relationship easier to visualise, to describe, and to compare.

Ladder of Transformations

- For the y-variable, consider transformations of the form

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

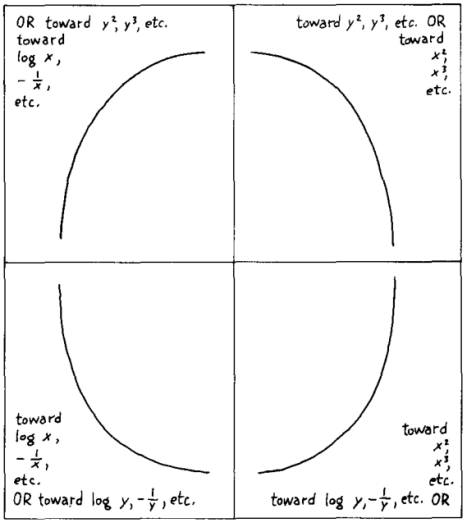
- Similarly, consider transformations of the form

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

  for the x-variable.
- When we begin, we are at the identity transformation.
- We shall try different transformations for each variable, and we refer to going to the right as "going up the ladder", and we refer to going to the left as "going down the ladder".

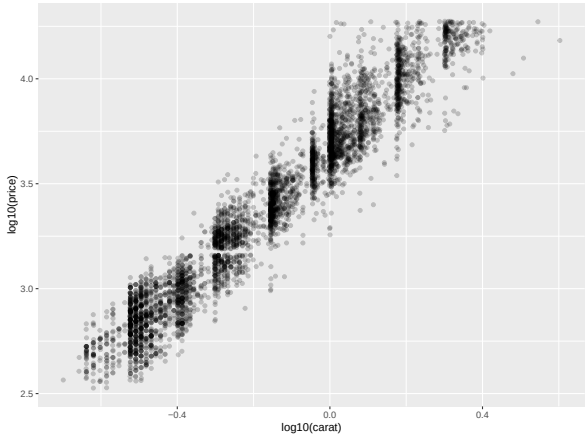
Ladder of Transformations

The following diagram guides us along which direction along the ladder to proceed.



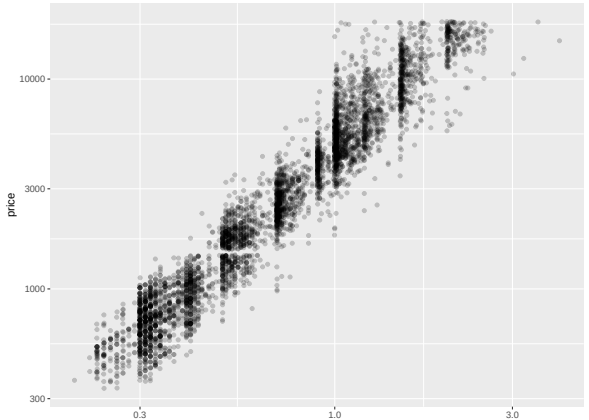
Diamond: Carat and Price

```
ggplot(sub_diamonds) +
  geom_point(aes(x=log10(carat), y=log10(price)),
             alpha=0.2, position="jitter")
```



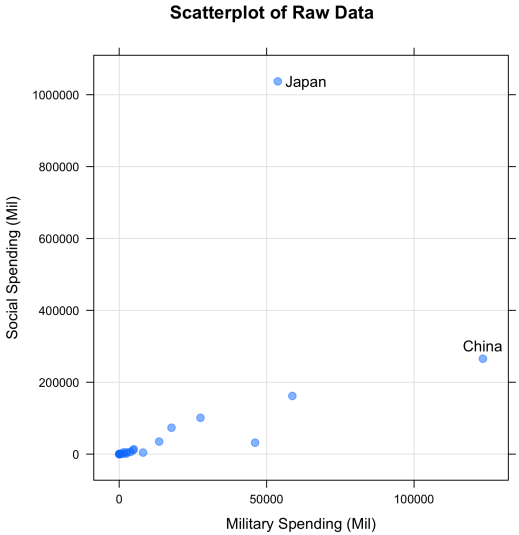
Better Scales

```
ggplot(sub_diamonds) +
  geom_point(aes(x=carat, y=price), alpha=0.2,
             position="jitter") +
  scale_x_log10() + scale_y_log10()
```



Military and Social Spending

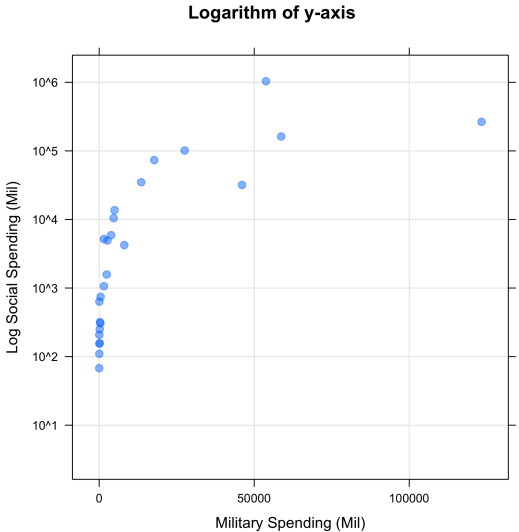
- Consider the following data.
- Social spending in Japan is very large compared to the rest, so we take logs to convert the y-axis to a multiplicative scale.



Military and Social Spending

cont'd

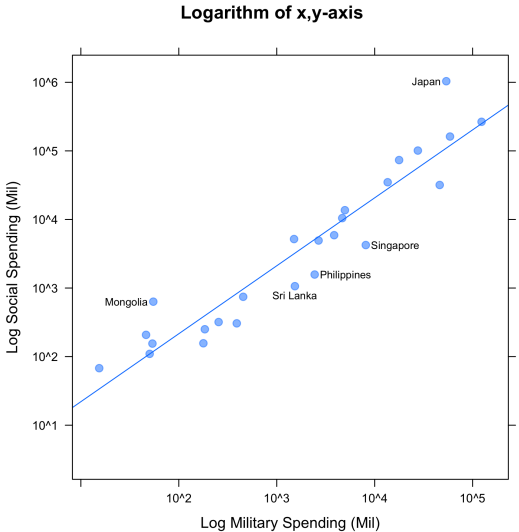
- After transforming the y-axis, we find that there is still room for improvement.



Military and Social Spending

cont'd

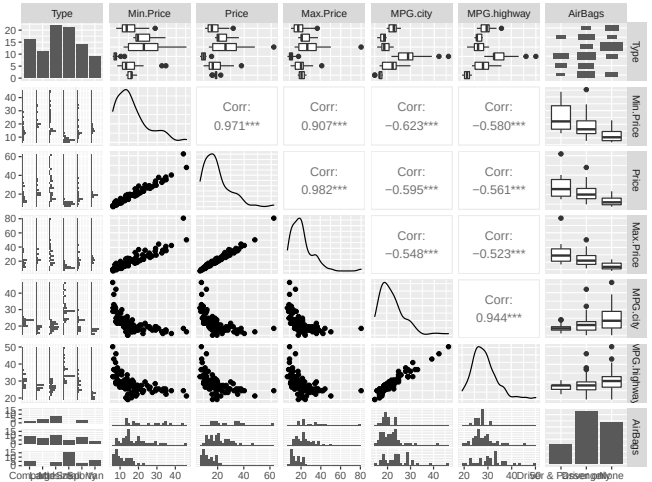
- This is the final plot.
- The relationship is simpler to describe.
- The labelled points are the ones that deviate the most from this relationship (largest residuals).



Pairs of Plots

- Plotting pair by pair is inefficient.
- Let's revisit the Cars93 dataset.

```
select(Cars93,
 `Type`:`AirBags`) %>%
  ggpairs
```



Summary

- Visualisation is a terrific tool for exploring data.
- Visualisation is an iterative procedure.
- The ability to ask and answer "What should I look at next?" is a critical part of data exploration.
- EDA can be very fun and rewarding.