Learning Objectives: Specialized Visualizations

- **Create a heat map**
- **Create a correlogram**
- **Create a mosaic plot**
- **Determine the best specialized chart to use based on the data provided**

definition

Assumptions

- Learners are comfortable reading and importing CSV data sets, extracting relevant data into data frames, and printing that data to the console.

Limitations

- This section will cover specialized visualizations in brief details only and will offer practical visualization functions for learners to start creating specialized charts right away.

Heat Maps

Creating Heat Maps

Follow the directions below to open up the `heatmap.r` file in RStudio.

info

Open the `heatmap.r` file

Within RStudio, open the `heatmap.r` file by selecting: `File` —> `Open File...` —> `code` —> `spec` —> `heatmap.r`

Data Import

```
# Some sector and geographies for illustration
sector <- c("Technology","Consumer Discretionary", "Real
        Estate", "Financial Services")
geography <- c("Asia", "Europe", "US", "India")
# Simulate market returns based on a random normal distribution,
        with a mean return of 8% and volatility (standard
        deviation) 15%
returns <- data.frame(Returns = round(rnorm(n = 1000, mean =
        0.08, sd = 0.15), 2))

mydata <- cbind(Sector = sample(sector, 1000, replace = T), Geo
        = sample(geography, 1000, replace = T), returns)

# Let's take a simple average across sector + geographies
myAvgRet <- mydata %>%
  group_by(Sector, Geo) %>%
  summarise(AVGreturns = mean(Returns))

myAvgRet.mat <- myAvgRet %>%
  # Convert long-form to wide-form
  spread(key = Geo, value = AVGreturns) %>%
  as.data.frame %>%
  # Extract column 'Sector' and use it to name rows.
  # This is necessary so the final output is a numeric matrix,
        the input which the heatmap function takes
  column_to_rownames(var = "Sector") %>%
  as.matrix
```

The basic syntax is:

```
ggplot($data, aes(x = $x, $y)) +
  geom_tile(aes(fill = AVGreturns))
```

Add on the following code into the text editor and then click the `Source` button to see the result.

```
plot <- ggplot(myAvgRet, aes(x = Sector, Geo)) +
  geom_tile(aes(fill = AVGreturns))
print(plot)
```

Correlograms

Creating Correlograms

Follow the directions below to open up the `correlogram.r` file in RStudio.

info

Open the `correlogram.r` file

Within RStudio, open the `correlogram.r` file by selecting: `File` —> `Open File...` —> `code` —> `spec` —> `correlogram.r`

Data Import

```
data(mtcars)
corr <- round(cor(mtcars), 1)
```

The basic syntax is:

```
ggcorrplot($data)
```

Add on the following code into the text editor and then click the `Source` button to see the result.

```
plot <- ggcorrplot(corr)
print(plot)
```

Mosaic Plots

Creating Mosaic Plots

Follow the directions below to open up the `mosaic.r` file in RStudio.

info

Open the `mosaic.r` file

Within RStudio, open the `mosaic.r` file by selecting: `File` —> `Open File...` —> `code` —> `spec` —> `mosaic.r`

Data Import

```
df = structure(list(effect = structure(c(2L, 2L, 1L, 1L), .Label
    = c("yes",
  "no"), class = "factor"), sex = structure(c(1L, 2L, 1L,
    2L),
   .Label = c("f",  "m"), class = "factor"), n = c(8, 3, 8,
    12)),
   row.names = c(NA, -4L), class = "data.frame")
```

The basic syntax is:

```
ggplot($data, aes(effect, y=$y, fill = $x)) +
  geom_col(position="fill")
```

Add on the following code into the text editor and then click the `Source` button to see the result.

```
plot <- ggplot(df, aes(effect, y=n, fill = sex)) +
  geom_col(position="fill")
print(plot)
```