Week 6 Module 4: Barplots

Exercises

Let's clear the global environment:

```
rm( list = ls() )
```

Now let's load in the Module 4 R objects:

```
load( "Module 4 R Objects.Rdata" )
```

Exercises

Exercise 1: Simple barplot

Here are the number of support calls for each of the regional offices of WiDgT:

Location	Number of support calls
Boston	43
London	27
Salt Lake City	11
Shanghai	56

Construct a barplot that displays the frequency counts of the number of support calls across the locations.

Solution

Exercise 2: Barplots and tables

Let's return to the `one.week.cereal.brand.character.string.vector` from the last module:

```
head( one.week.cereal.brand.character.string.vector )
```

```
## [1] "Krispee Yummm!!"       "Krispee Yummm!!"       "Sugar Bomz"
## [4] "Sugar Bomz"            "Healthy Kale and Tofu" "Krispee Yummm!!"
```

First, construct a table of the frequency counts.

Then create a properly formatted barplot for this table.

Finally, construct a table of relative frequencies for this data, and make another properly formatted barplot.

Solution

Exercise 3: Barplots and factors

In the previous exercise, we created a barplot of the frequency counts for the one-week cereal sales.

However, the order of the categories wasn't our usual one, and it would be nice to display the data with our standard conventions.

First, construct a factor where the levels are ordered with Sugar Bomz first, then Krispee Yummm!!, and finally Healthy Kale and Tofu. Also, change the labels of the levels so that they are "SBZ", "KYM", and "HKT", respectively.

Next, construct a frequency count table using this factor, and save this in a variable.

Then create a properly formatted barplot for this table.

Solution

Solutions to the Exercises

Exercise 1: Simple barplot

Here are the number of support calls for each of the regional offices of WiDgT:

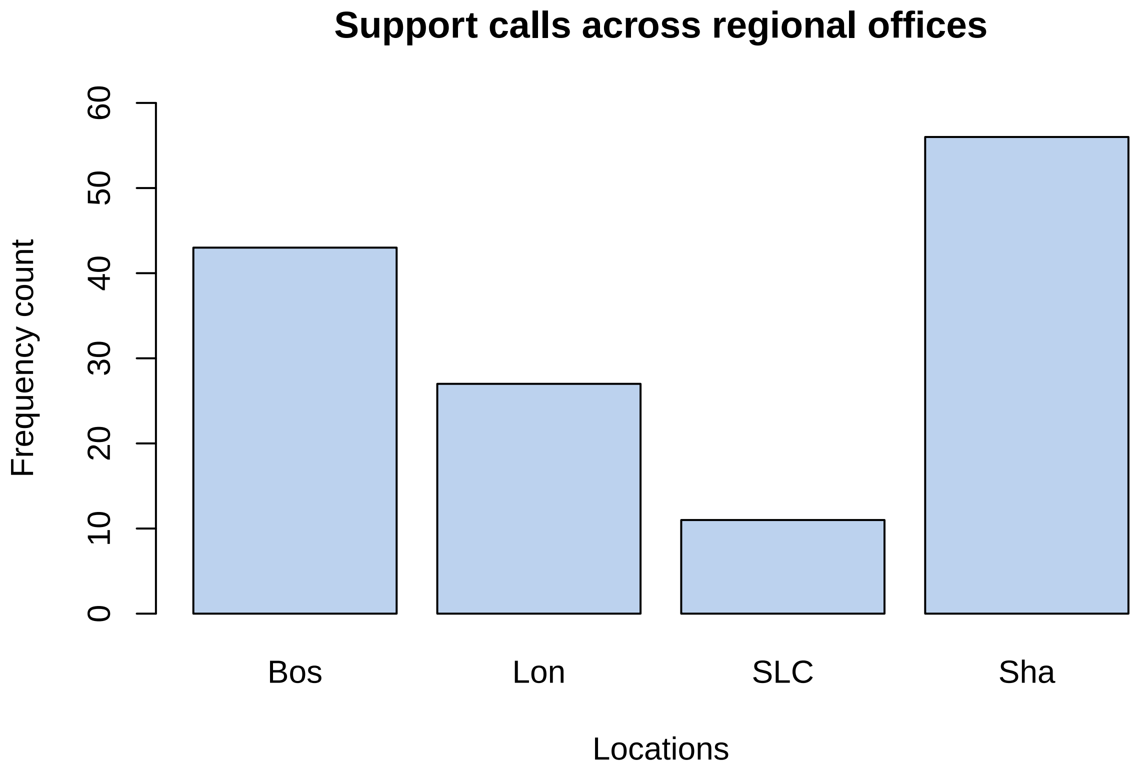
Location	Number of support calls
Boston	43
London	27
Salt Lake City	11
Shanghai	56

Construct a barplot that displays the frequency counts of the number of support calls across the locations.

Solution

```
support.calls.vector <-
  c( 43, 27, 11, 56 )

barplot(
  height = support.calls.vector,
  ylim = c(0, 60),
  main = "Support calls across regional offices",
  xlab = "Locations",
  ylab = "Frequency count",
  col = "lightsteelblue2",
  names.arg = c( "Bos", "Lon", "SLC", "Sha" ),
)
```



Exercise 2: Barplots and tables

Let's return to the `one.week.cereal.brand.character.string.vector` from the last module:

```
head( one.week.cereal.brand.character.string.vector )
```

```
## [1] "Krispee Yummm!!"       "Krispee Yummm!!"       "Sugar Bomz"
## [4] "Sugar Bomz"            "Healthy Kale and Tofu" "Krispee Yummm!!"
```

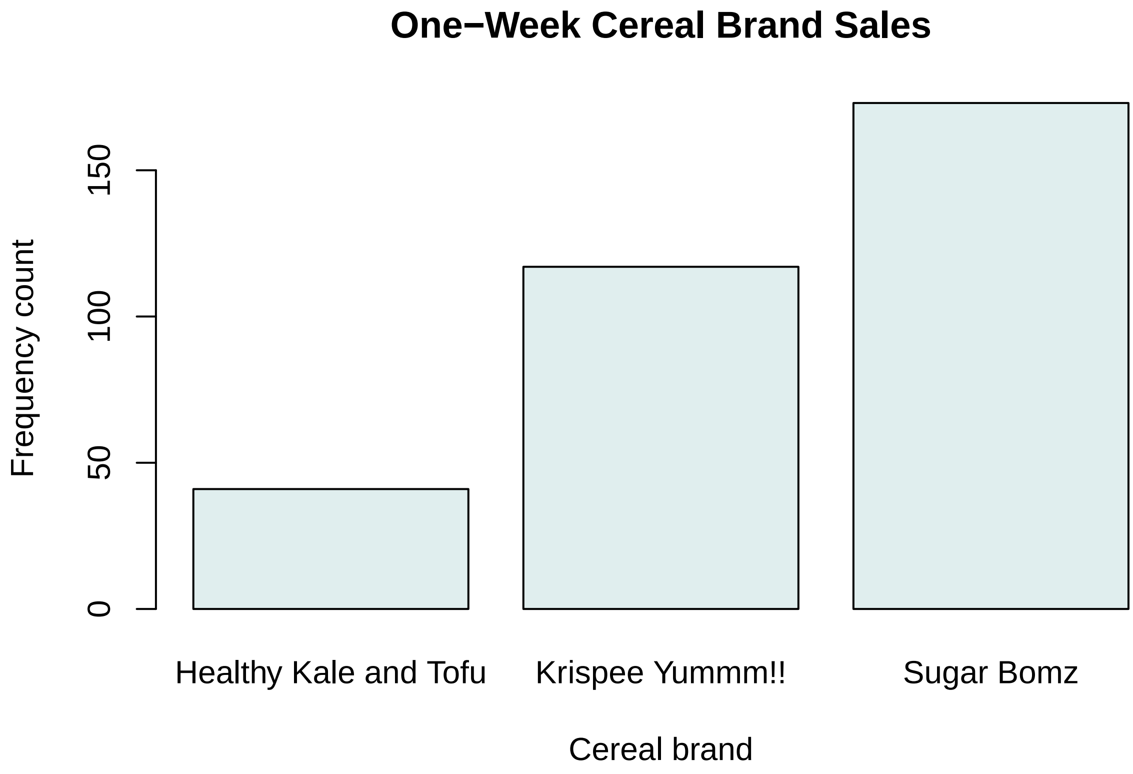
First, construct a table of the frequency counts for this data.

Then create a properly formatted barplot for this table.

Solution

```
cereal.brand.frequency.count.table <-
  table( one.week.cereal.brand.character.string.vector )

barplot(
  height = cereal.brand.frequency.count.table,
  main = "One-Week Cereal Brand Sales",
  xlab = "Cereal brand",
  ylab = "Frequency count",
  col = "azure2"
)
```



Exercise 3: Barplots and factors

In the previous exercise, we created a barplot of the frequency counts for the one-week cereal sales.

However, the order of the categories wasn't our usual one, and it would be nice to display the data with our standard conventions.

First, construct a factor where the levels are ordered with Sugar Bomz first, then Krispee Yummm!!, and finally Healthy Kale and Tofu. Also, change the labels of the levels so that they are "SBZ", "KYM", and "HKT", respectively.

Next, construct a frequency count table using this factor, and save this in a variable.

Then create a properly formatted barplot for this table.

Solution

First, we'll construct the factor:

```
one.week.cereal.brand.factor <-
  factor(
    x = one.week.cereal.brand.character.string.vector,
    levels = c( "Sugar Bomz", "Krispee Yummm!!", "Healthy Kale and Tofu" ),
    labels = c( "SBZ", "KYM", "HKT" )
  )
```

Next, we'll create the frequency count table, and save this in a variable:

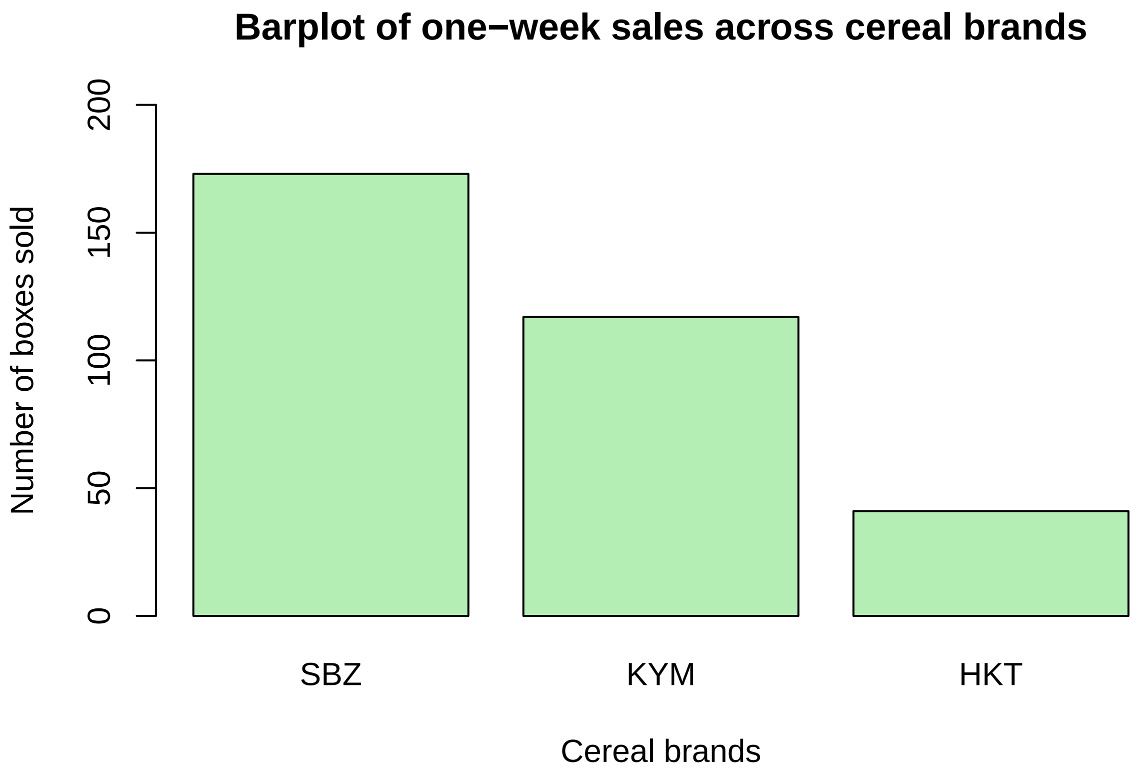
```
one.week.cereal.brand.frequency.count.table <-
  table( one.week.cereal.brand.factor )

one.week.cereal.brand.frequency.count.table
```

```
## one.week.cereal.brand.factor
## SBZ KYM HKT
## 173 117  41
```

Now we can create the barplot:

```
barplot(
  height = one.week.cereal.brand.frequency.count.table,
  ylim = c(0, 200),
  main = "Barplot of one-week sales across cereal brands",
  xlab = "Cereal brands",
  ylab = "Number of boxes sold",
  col = "darkseagreen2"
)
```


Barplot of one−week sales across cereal brands
Number of boxes sold
SBZ
KYM
HKT
Cereal brands