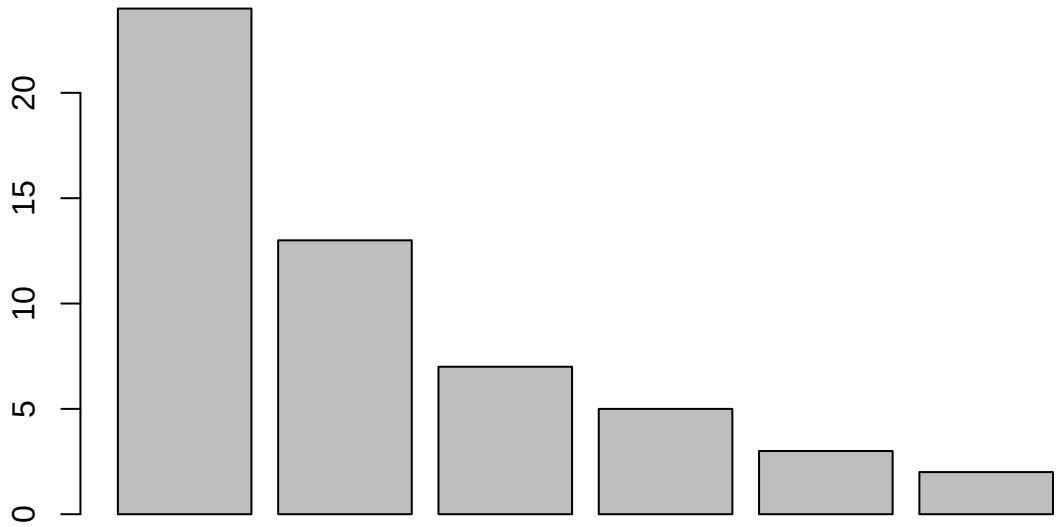
Data Visualization in R

Howard Nguyen

2020-01-02

LOAD DATA

```
x = c(24, 13, 7, 5, 3, 2)  # Sample data
barplot(x)                 # Default barplot
```



COLORS IN R

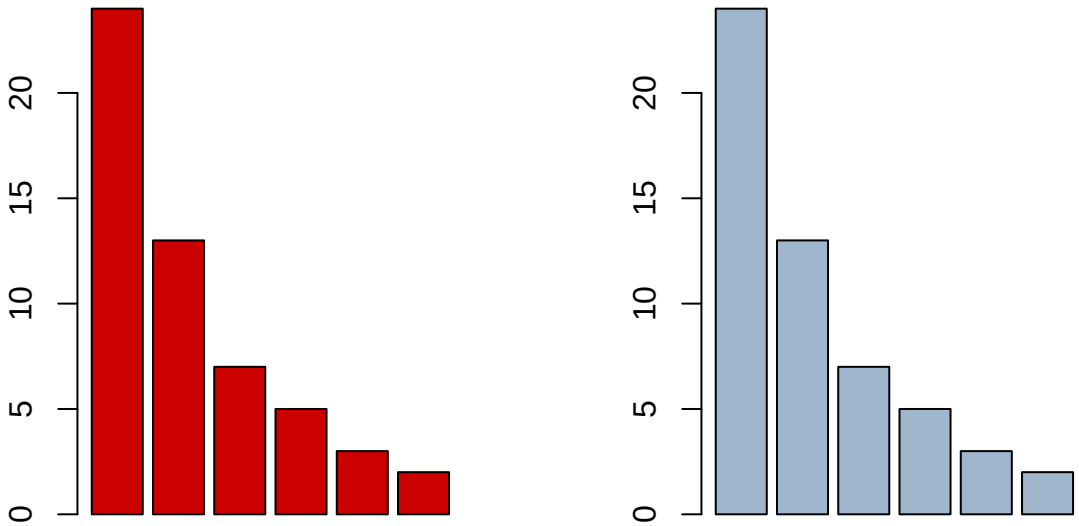
```
# Color names R has 657 color names for 502 unique colors,
# arranged alphabetically except for white, which is first
```

```
# ?colors
# colors()  # Get list of color names
```

```
# Web page with R colors swatches, color names, hex codes,
# RBG codes (in 0-255 and 0.00-1.00), and R index numbers;
# Browsable table on the page or in Google Sheets;
# downloadable as XLSX or PDF
# browseURL("https://datalab.cc/rcolors")
```

USE COLORS

```
par(mfrow=c(1,2))
# Color names
barplot(x, col = "red3")  # red3
barplot(x, col = "slategray3")  # slategray3
```

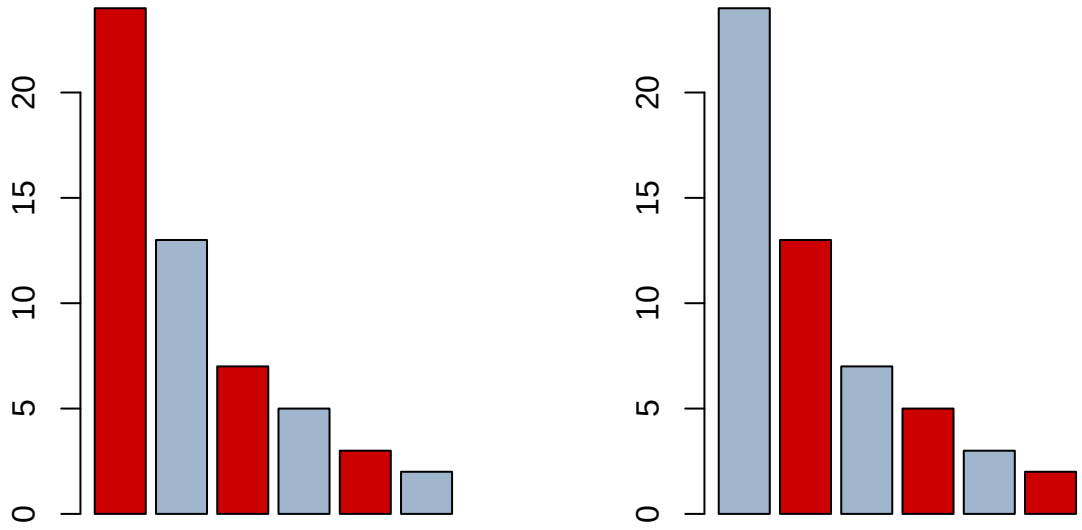



```
# RGB triplets (0.00-1.00)
#barplot(x, col = rgb(.80, 0, 0))      # red3
#barplot(x, col = rgb(.62, .71, .80))  # slategray3
# RGB triplets (0-255)
#barplot(x, col = rgb(205, 0, 0, max = 255))     # red3
#barplot(x, col = rgb(159, 182, 205, max = 255)) # slategray3
```

```
# RGB hexcodes
#barplot(x, col = "#CD0000")  # red3
#barplot(x, col = "#9FB6CD")  # slategray3
# Index numbers
#barplot(x, col = colors() [555])  # red3
#barplot(x, col = colors() [602])  # slategray3
```

MULTIPLE COLORS

```
par(mfrow=c(1,2))
# Can specify several colors in a vector, which will cycle
barplot(x, col = c("red3", "slategray3"))
barplot(x, col = c("#9FB6CD", "#CD0000"))
```

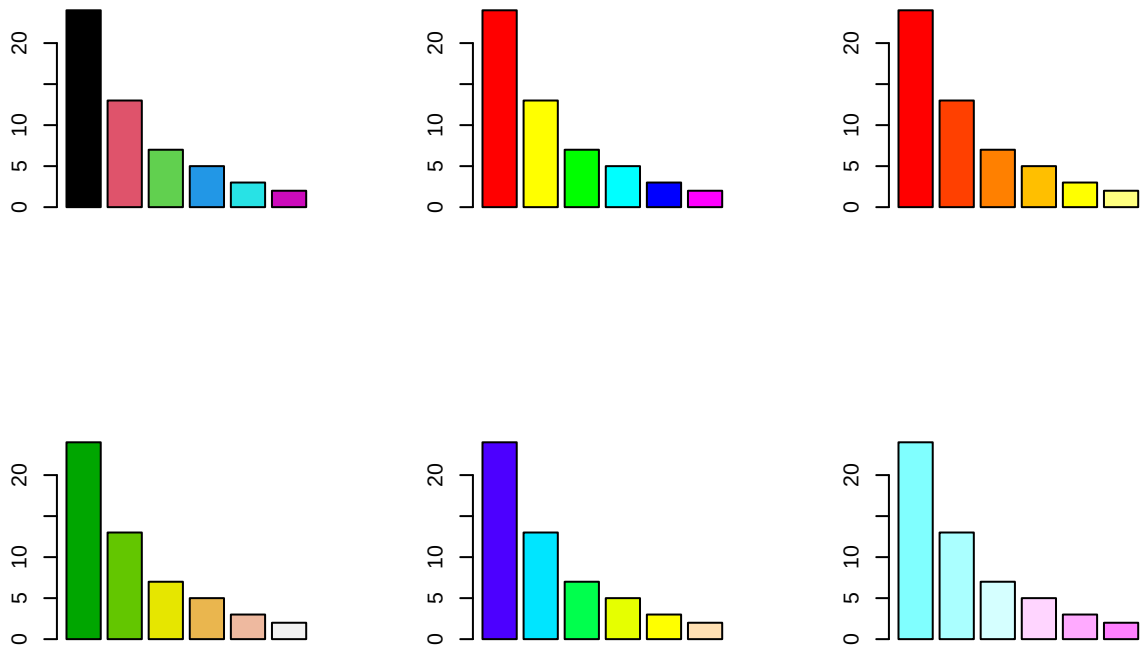


USING COLOR PALETTES

```
?palette  # Info on palettes
palette() # See current palette
```

```
## [1] "black"   "#DF536B" "#61D04F" "#2297E6" "#28E2E5" "#CD0BBC" "#F5C710"
## [8] "gray62"
```

```
par(mfrow=c(2,3))
barplot(x, col = 1:6)                # Use current palette
barplot(x, col = rainbow(6))         # Rainbow colors
barplot(x, col = heat.colors(6))     # Yellow through red
barplot(x, col = terrain.colors(6))  # Gray through green
barplot(x, col = topo.colors(6))     # Purple through tan
barplot(x, col = cm.colors(6))       # Pinks and blues
```



LOAD PACKAGES

RStudio will prompt you to download any packages that aren't already installed.

```
# Load packages
library(tidyverse)  # Loads the `tidyverse` collection
library(readxl)     # Reads CSV and Excel files
```

LOAD DATA

```
# Also convert several adjacent variables to factors
df <- read_csv("../data/state_trends.csv") |>
  mutate(across(c(region:psy_reg), factor)) |>
  print()
```

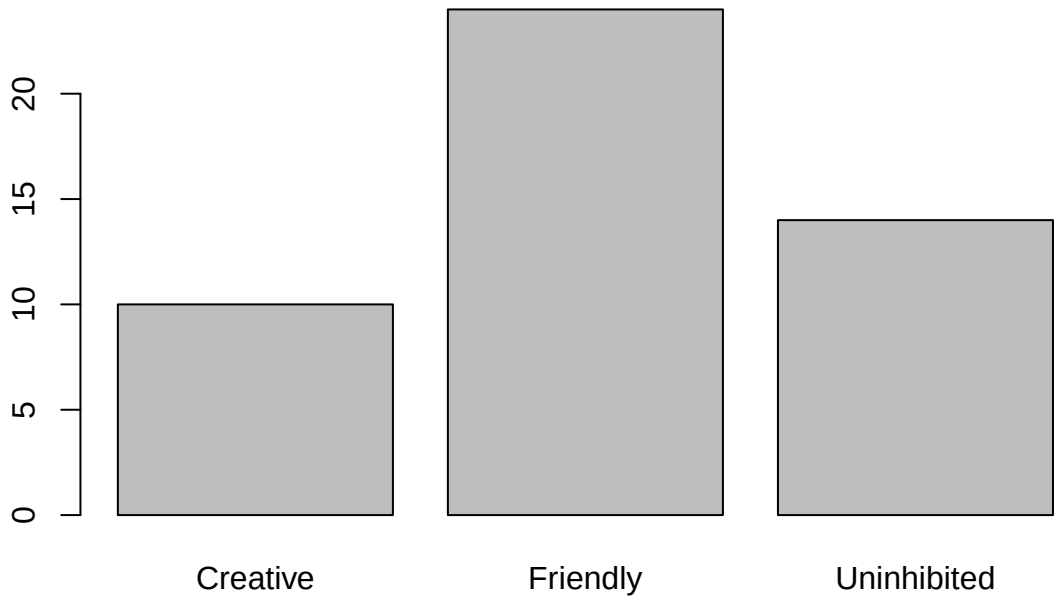
```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 34
##    state  state~1 popul~2 sq_mi~3 pop_d~4 region psych~5 psy_reg extra~6 agree~7
##    <chr>  <chr>     <dbl>   <dbl>   <dbl> <fct>  <fct>   <fct>     <dbl>   <dbl>
##  1 Alaba~ AL       5.02e6   52420      96 South  Friend~ Friend~    55.5    52.7
##  2 Arizo~ AZ       7.15e6  113990      63 West   Relaxe~ Creati~    50.6    46.6
##  3 Arkan~ AR       3.01e6   53179      57 South  Friend~ Friend~    49.9    52.7
##  4 Calif~ CA       3.95e7  163695     242 West   Relaxe~ Creati~    51.4    49
##  5 Color~ CO       5.77e6  104094      55 West   Friend~ Friend~    45.3    47.5
##  6 Conne~ CT       3.61e6    5543     650 North~ Temper~ Uninhi~    57.6    38.6
##  7 Delaw~ DE       9.90e5    2489     398 South  Temper~ Uninhi~    47      38.8
##  8 Flori~ FL       2.15e7   65758     328 South  Friend~ Friend~    60.9    50.7
##  9 Georg~ GA       1.07e7   59425     180 South  Friend~ Friend~    63.2    60
## 10 Idaho  ID       1.84e6   83569      22 West   Relaxe~ Creati~    40.7    52.9
## # ... with 38 more rows, 24 more variables: conscientiousness <dbl>,
## #   neuroticism <dbl>, openness <dbl>, data_science <dbl>,
## #   artificial_intelligence <dbl>, machine_learning <dbl>, data_analysis <dbl>,
## #   business_intelligence <dbl>, spreadsheet <dbl>, statistics <dbl>,
## #   art <dbl>, dance <dbl>, museum <dbl>, basketball <dbl>, football <dbl>,
## #   baseball <dbl>, soccer <dbl>, hockey <dbl>, has_nba <chr>, has_nfl <chr>,
## #   has_mlb <chr>, has_mls <chr>, has_nhl <chr>, has_any <chr>, and ...
```

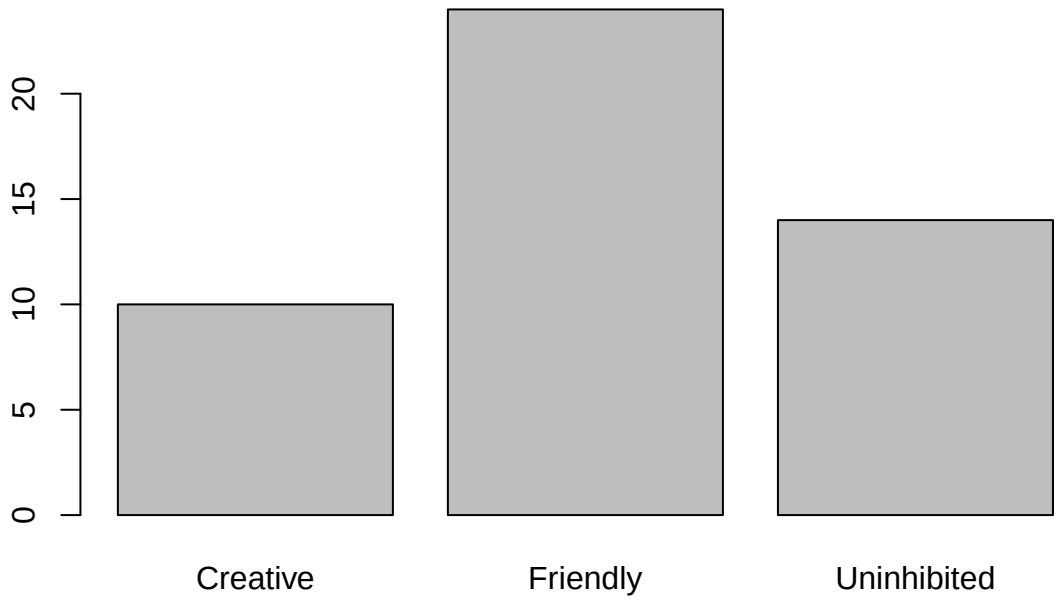
BARPLOT OF FREQUENCIES

?plot # Get info on “Generic X-Y Plotting ?barplot # Get info on the”Bar Plots” function

```
# Shortest method to make a barplot
plot(df$psy_reg)
```

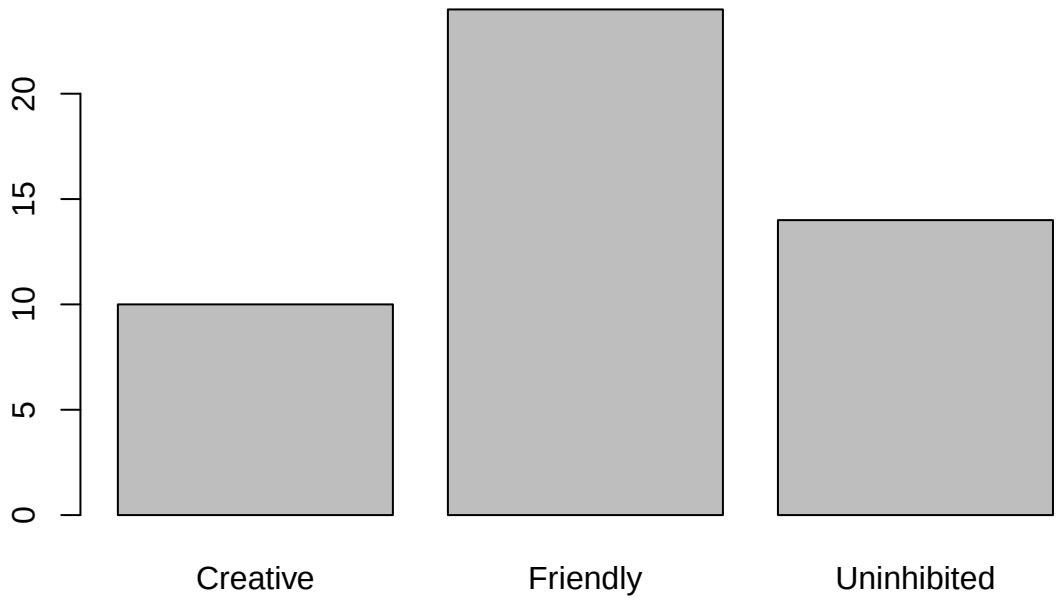



```
# Similar process using pipes
df |>
  select(psy_reg) |>
  plot()
```

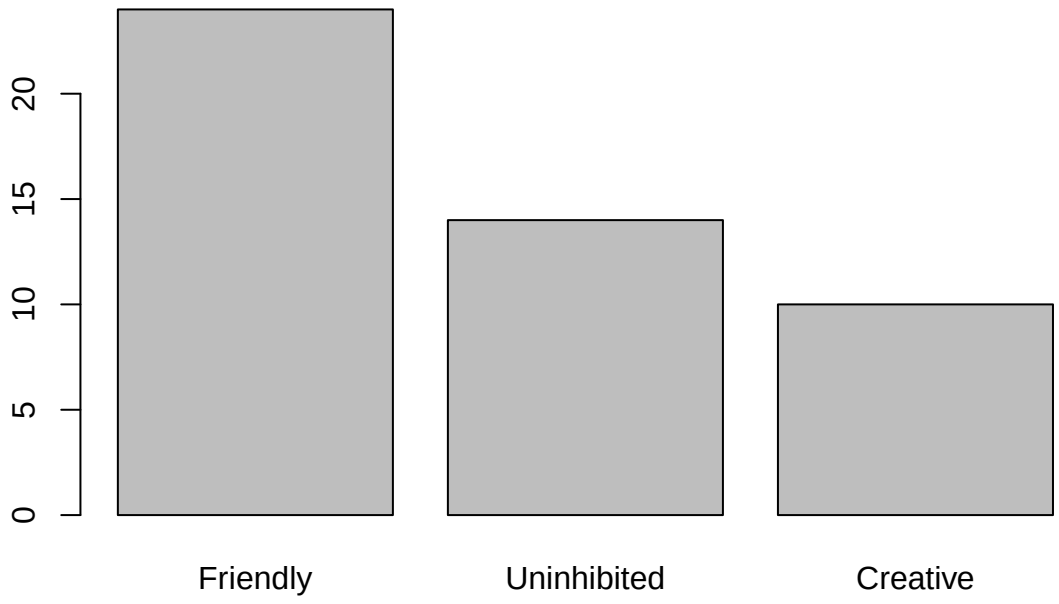



```
# Similar code using barplot(); DOESN'T WORK
#df |>
#  select(psy_reg) |>
#  barplot()  # Error: height must be a vector or a matrix
```

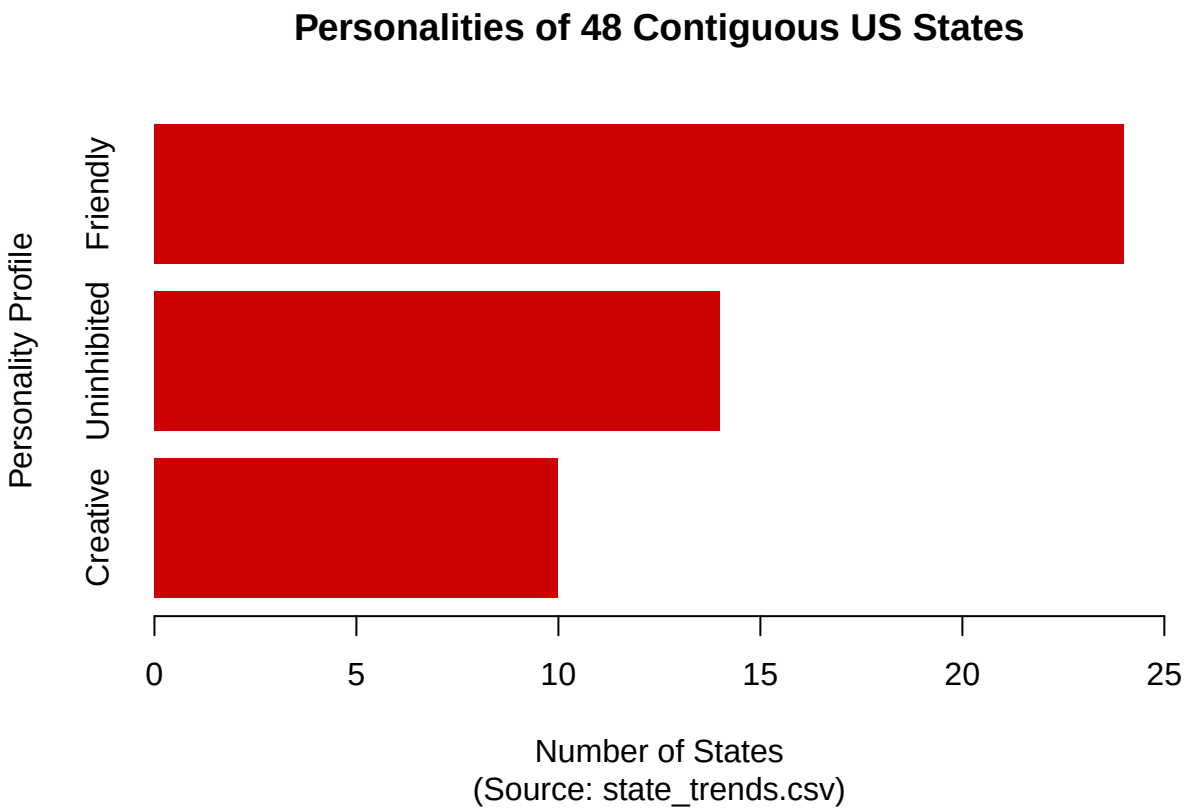
```
# Create frequency vector with table()
df |>
  select(psy_reg) |>
  table() |>  # Put data in appropriate format
  barplot()
```




```
# Sort bars by decreasing values (NOT for ordinal X)
df |>
  select(psy_reg) |>
  table() |>
  sort(decreasing = T) |>  # Sort table
  barplot()
```

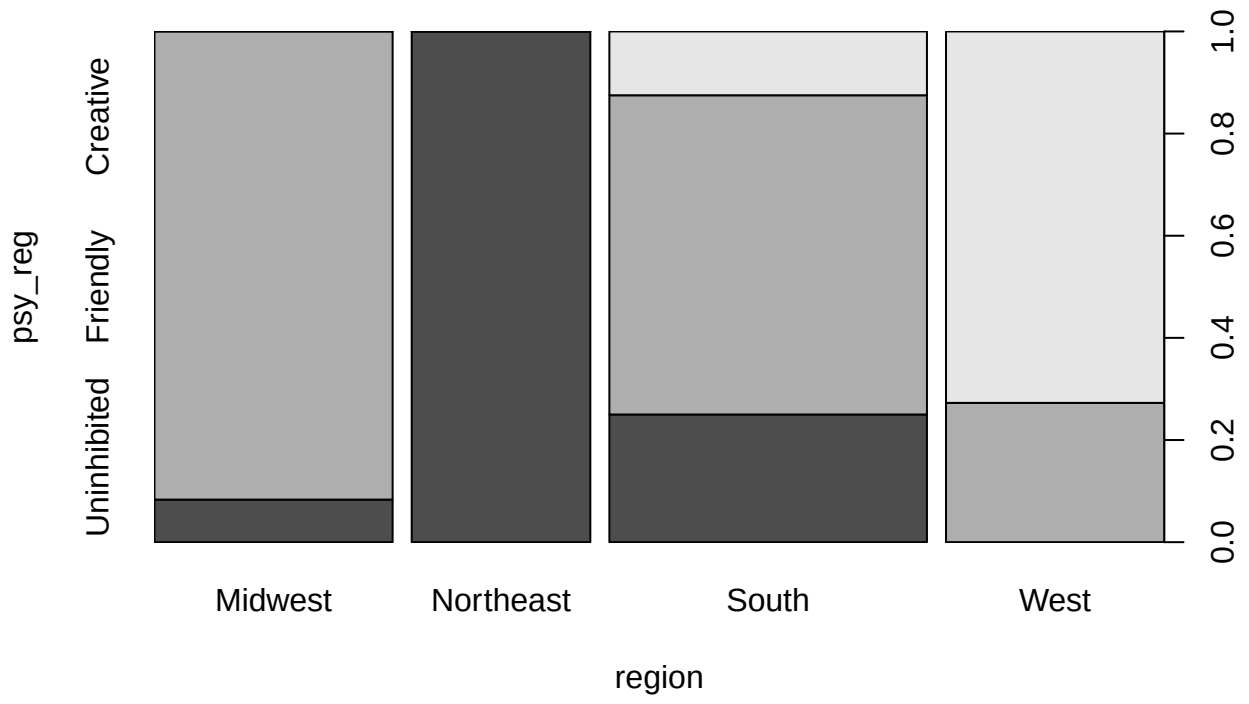



```r
# Add options to plot
df |>
  select(psy_reg) |>
  table() |>  # Put data in appropriate format
  sort(decreasing = F) |>  # Sort table
  barplot(
    main   = "Personalities of 48 Contiguous US States",
    sub    = "(Source: state_trends.csv)",
    horiz  = T,  # Draw horizontal bars
    ylab   = "Personality Profile",
    xlab   = "Number of States",
    xlim   = c(0, 25),  # Limits for X axis
    border = NA,  # No borders on bars
    col    = "#CD0000"  # red3
  )
```



STACKED BARPLOT OF FREQUENCIES

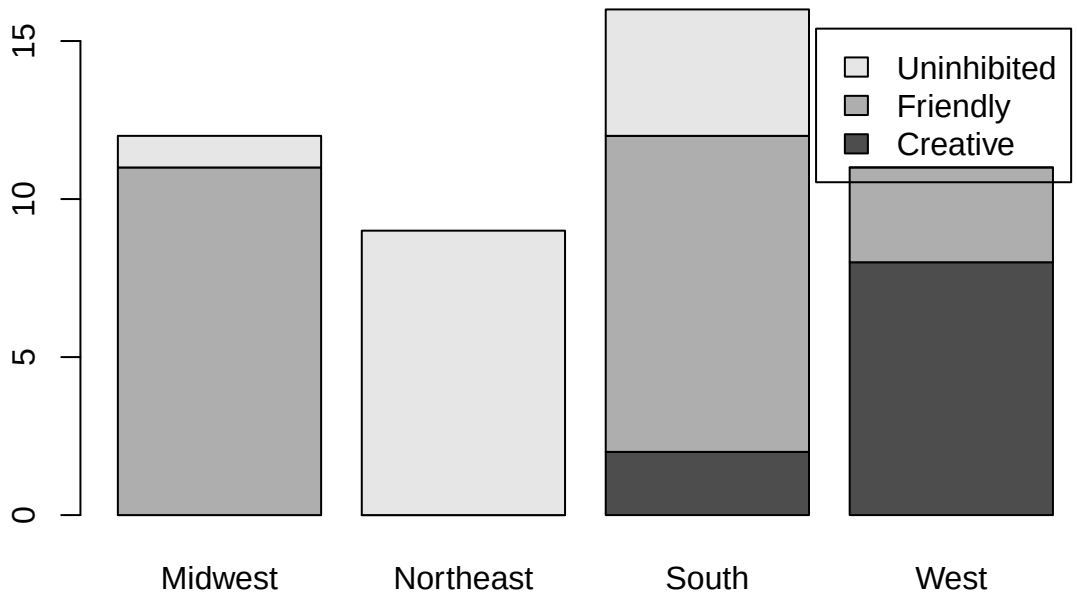
```
# 100% stacked bar
df |>
  select(region, psy_reg) |>
  plot()
```



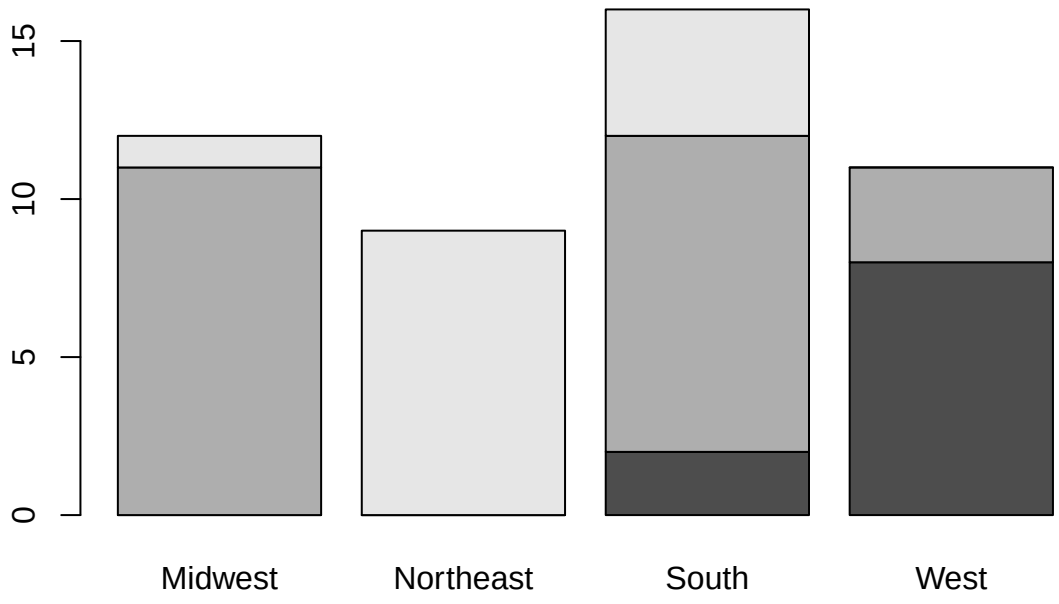

```
# Stacked bars: step 1: create table
df_t <- df |>
  select(psy_reg, region) |>
  table() |>
  print()  # Show table in Console
```

```
##              region
## psy_reg       Midwest Northeast South West
##   Creative          0         0     2    8
##   Friendly         11         0    10    3
##   Uninhibited       1         9     4    0
```

```
# Stacked bars: step 2a: create graph w/legend
df_t |> barplot(legend = rownames(df_t))
```

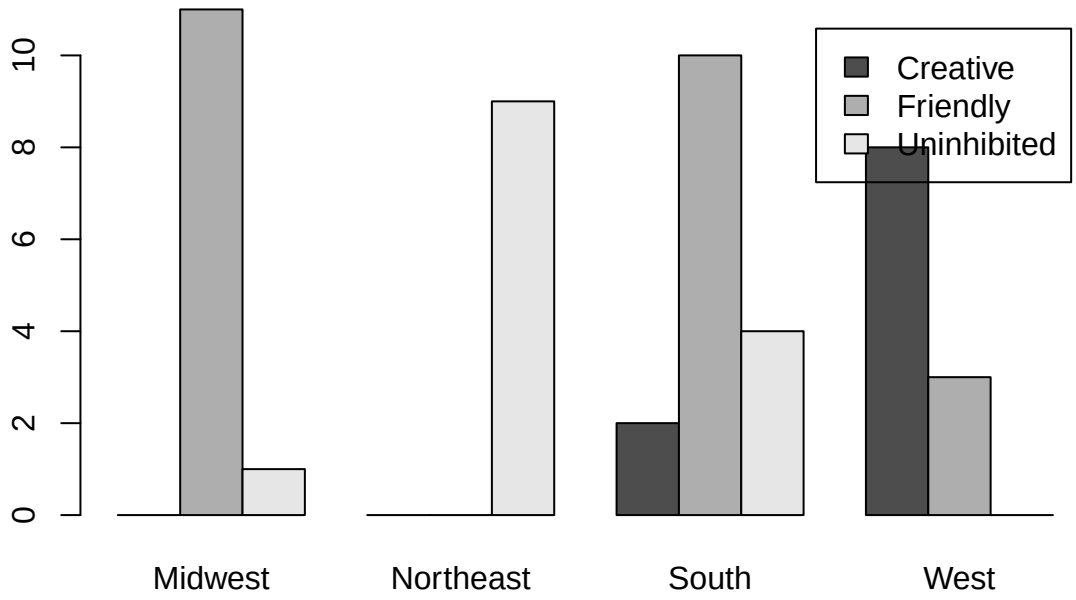



```
# Stacked bars: step 2b: create graph w/o legend
df_t |> barplot()
```

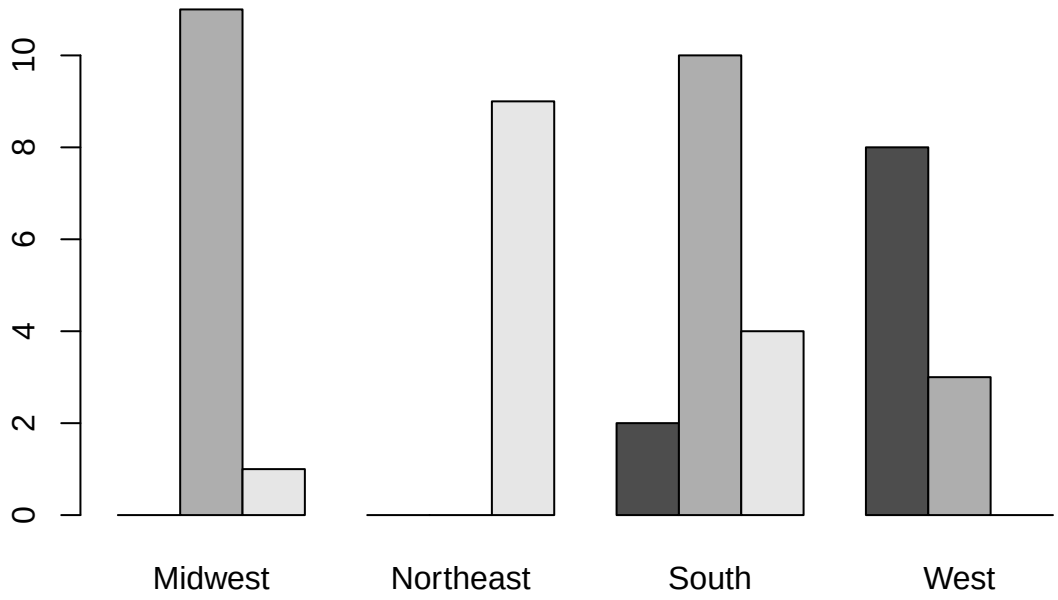


SIDE-BY-SIDE BARPLOT OF FREQUENCIES

```
# Side-by-side bar w/legend
df_t |>
  barplot(
    legend = rownames(df_t),
    beside = T  # Put bars next to each other
  )
```

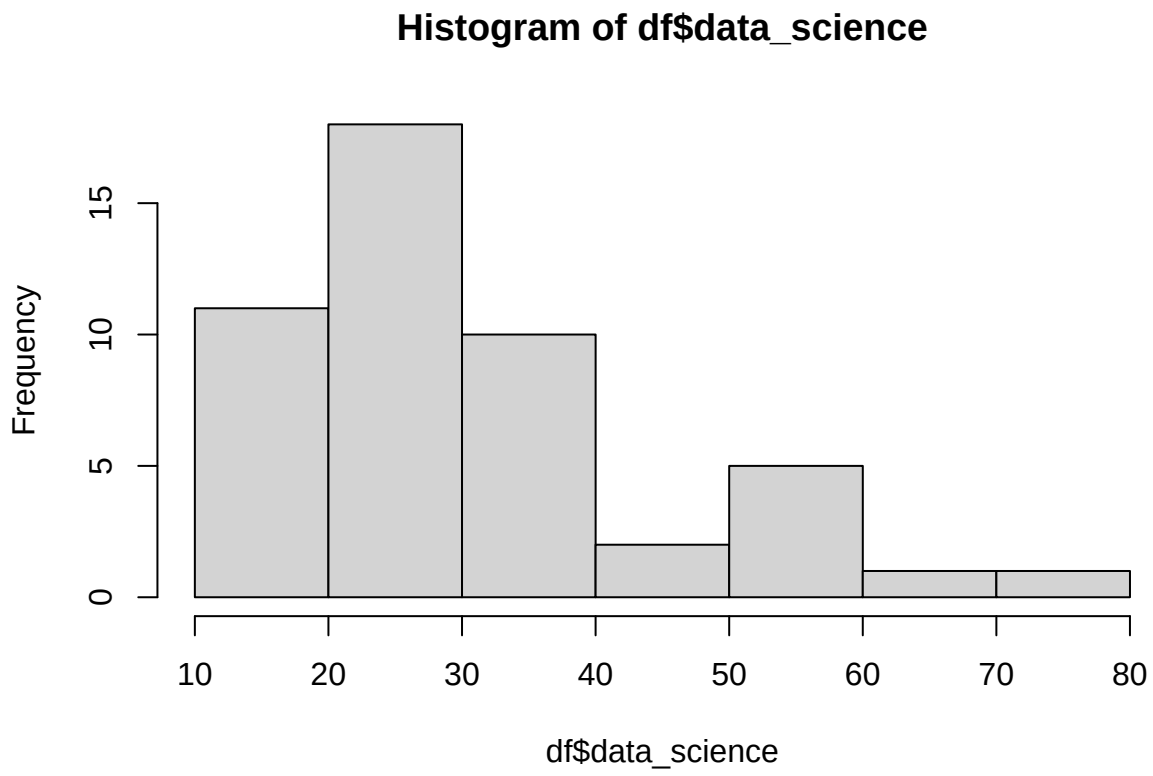



```
# Side-by-side bar w/o legend
df_t |> barplot(beside = T)
```

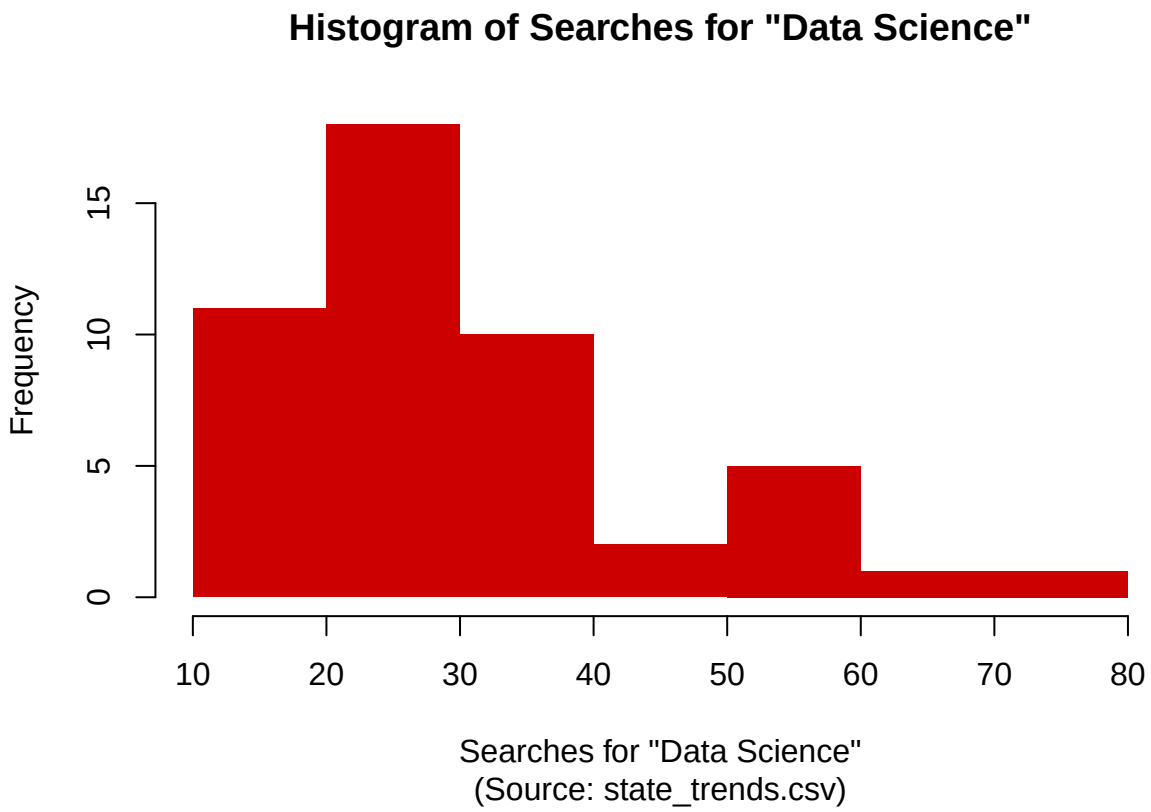


HISTOGRAM

```
# Histogram with defaults
hist(df$data_science)
```

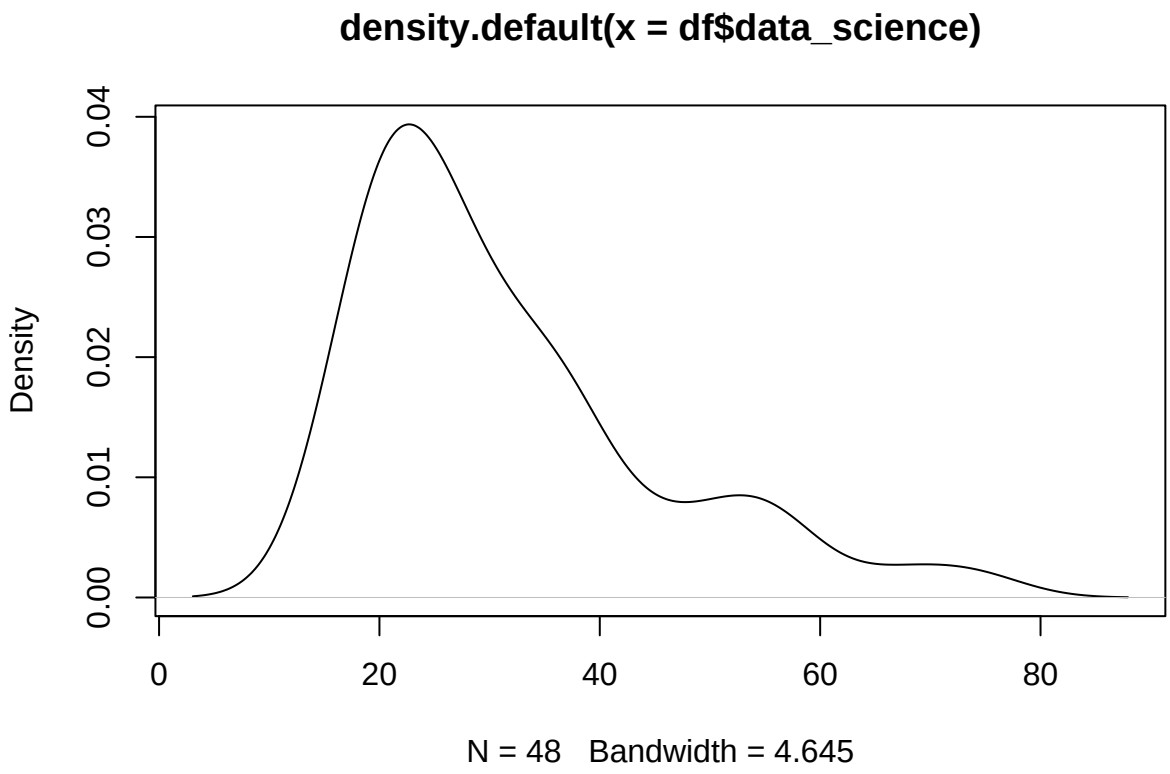



```
# Histogram with options
hist(df$data_science,
  breaks = 7,  # Suggest number of breaks
  main   = "Histogram of Searches for \"Data Science\"",
  sub    = "(Source: state_trends.csv)",
  ylab   = "Frequency",
  xlab   = "Searches for \"Data Science\"",
  border = NA,  # No borders on bars
  col    = "#CD0000"  # Sets fill color to red3
)
```

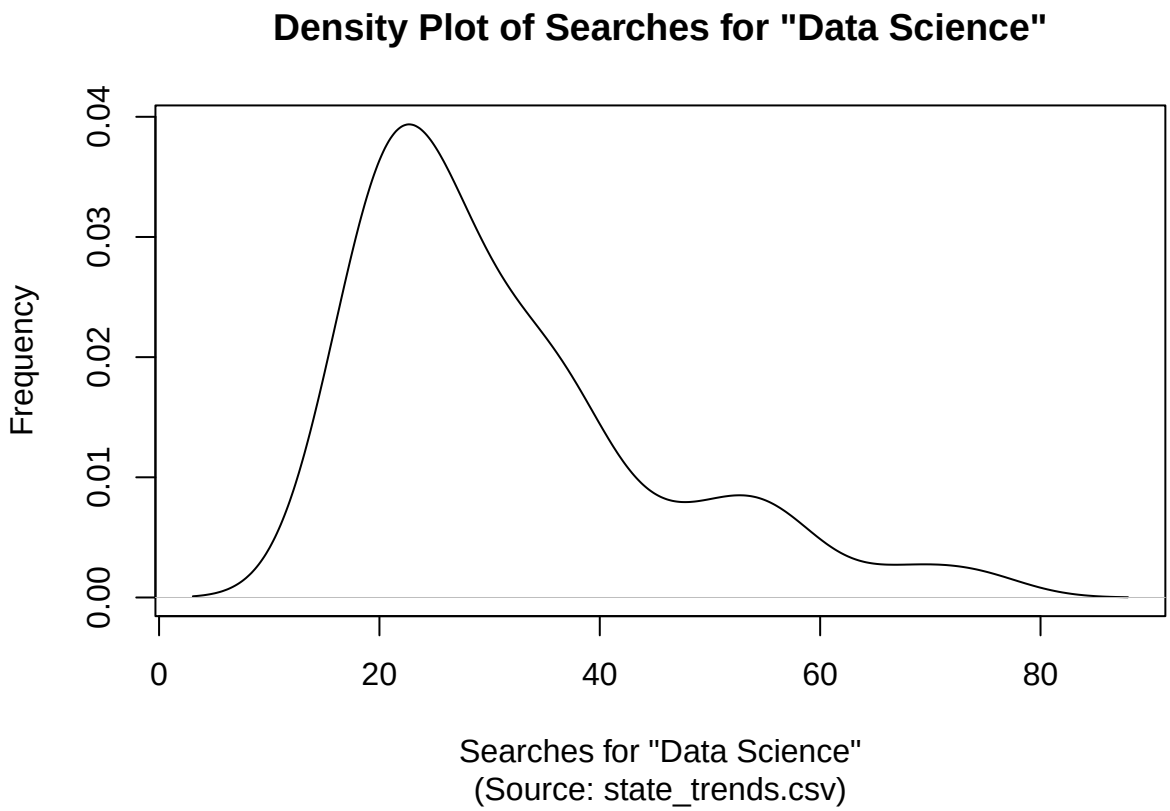


DENSITY PLOT

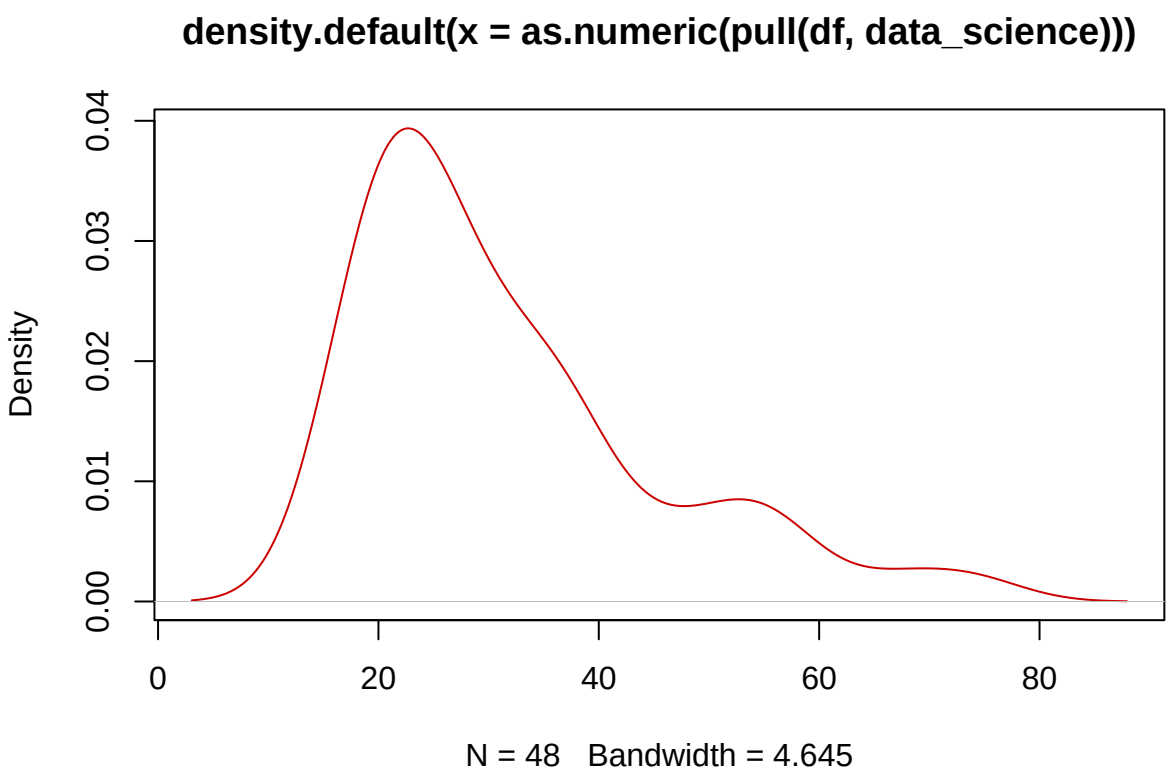
```
# Density plot with defaults
plot(density(df$data_science))
```




```
# Density plot with options
df |>
  pull(data_science) |>  # Use pull() instead of select()
  as.numeric() |>        # Coerces to numeric variable
  density() |>           # Draws density curve
  plot(
    main = "Density Plot of Searches for \"Data Science\"",
    sub  = "(Source: state_trends.csv)",
    ylab = "Frequency",
    xlab = "Searches for \"Data Science\"",
  )
```




```
# Use polygon to ADD a filled density plot
#plot.new()
#plot(density(df$data_science))
df |>
  pull(data_science) |>
  as.numeric() |>
  density() |>
  plot(col  = "#CD0000")  # Sets fill color to red3
```



BOXPLOT OF FREQUENCIES

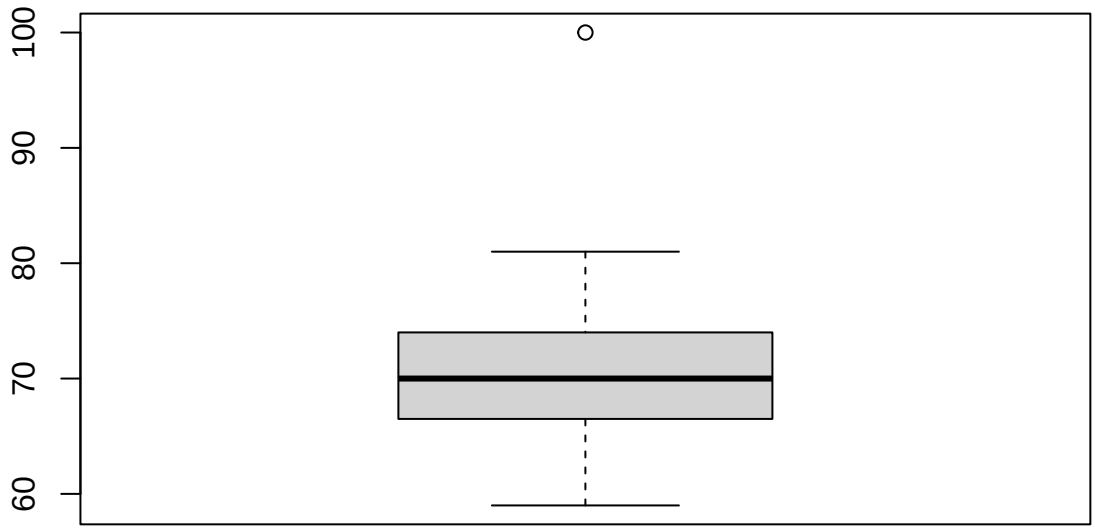
```
# Also convert all character variables to factors
df <- read_csv("../data/state_trends.csv") |>
  mutate(across(where(is_character), as_factor)) |>
  print()
```

```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 34
##    state  state~1 popul~2 sq_mi~3 pop_d~4 region psych~5 psy_reg extra~6 agree~7
##    <fct>  <fct>     <dbl>   <dbl>   <dbl> <fct>  <fct>   <fct>     <dbl>   <dbl>
##  1 Alaba~ AL       5.02e6   52420      96 South  Friend~ Friend~    55.5    52.7
##  2 Arizo~ AZ       7.15e6  113990      63 West   Relaxe~ Creati~    50.6    46.6
##  3 Arkan~ AR       3.01e6   53179      57 South  Friend~ Friend~    49.9    52.7
```

```
##  4 Calif~ CA       3.95e7  163695     242 West   Relaxe~ Creati~    51.4    49
##  5 Color~ CO       5.77e6  104094      55 West   Friend~ Friend~    45.3    47.5
##  6 Conne~ CT       3.61e6    5543     650 North~ Temper~ Uninhi~    57.6    38.6
##  7 Delaw~ DE       9.90e5    2489     398 South  Temper~ Uninhi~    47      38.8
##  8 Flori~ FL       2.15e7   65758     328 South  Friend~ Friend~    60.9    50.7
##  9 Georg~ GA       1.07e7   59425     180 South  Friend~ Friend~    63.2    60
## 10 Idaho  ID       1.84e6   83569      22 West   Relaxe~ Creati~    40.7    52.9
## # ... with 38 more rows, 24 more variables: conscientiousness <dbl>,
## #   neuroticism <dbl>, openness <dbl>, data_science <dbl>,
## #   artificial_intelligence <dbl>, machine_learning <dbl>, data_analysis <dbl>,
## #   business_intelligence <dbl>, spreadsheet <dbl>, statistics <dbl>,
## #   art <dbl>, dance <dbl>, museum <dbl>, basketball <dbl>, football <dbl>,
## #   baseball <dbl>, soccer <dbl>, hockey <dbl>, has_nba <fct>, has_nfl <fct>,
## #   has_mlb <fct>, has_mls <fct>, has_nhl <fct>, has_any <fct>, and ...
```

```
# Boxplot with defaults
boxplot(df$dance)
```

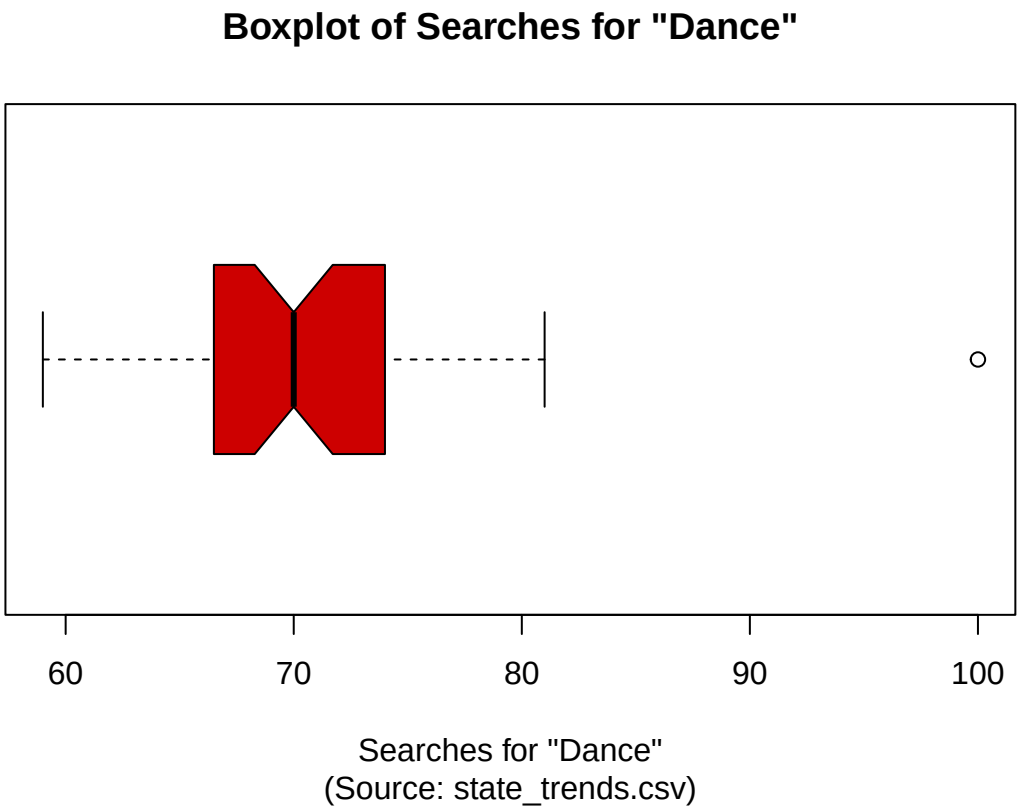



```
# Who is the outlier?
df |>
  filter(dance > 90) |>
  select(state, dance)
```

```
## # A tibble: 1 x 2
##   state dance
```

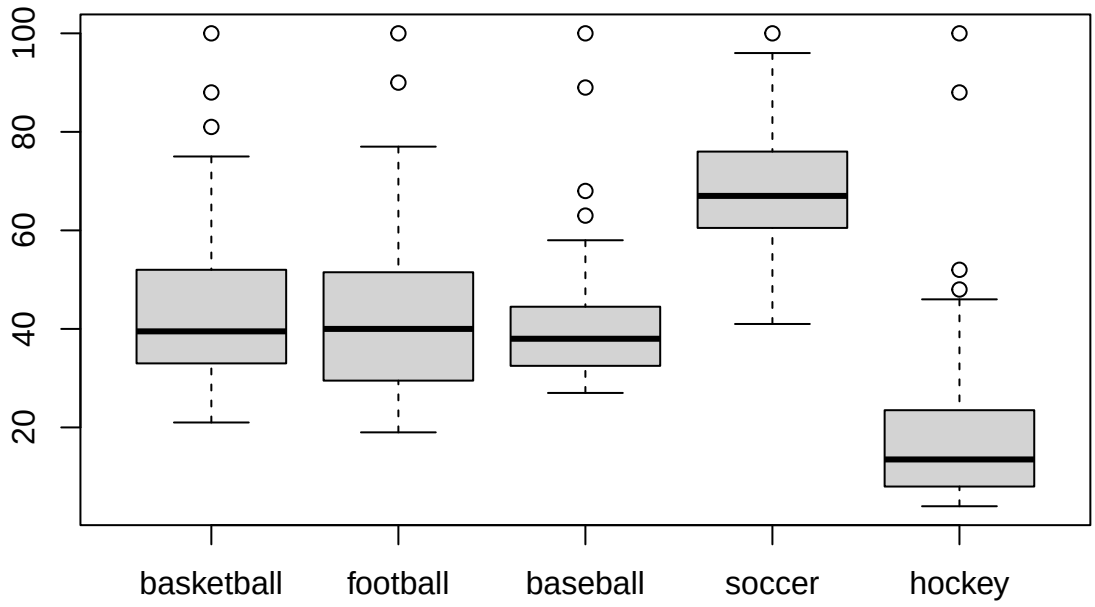
```
##   <fct> <dbl>
## 1 Utah    100
```

```
# Boxplot with options
df |>
  select(dance) |>
  boxplot(
    horizontal = T,  # Horizontal
    notch  = T,      # Confidence interval for median
    main   = "Boxplot of Searches for \"Dance\"",
    sub    = "(Source: state_trends.csv)",
    xlab   = "Searches for \"Dance\"",
    col    = "#CD0000"  # red3
  )
```



BOXPLOTS FOR MULTIPLE VARIABLES

```
df |>
  select(basketball:hockey) |>
  boxplot()
```

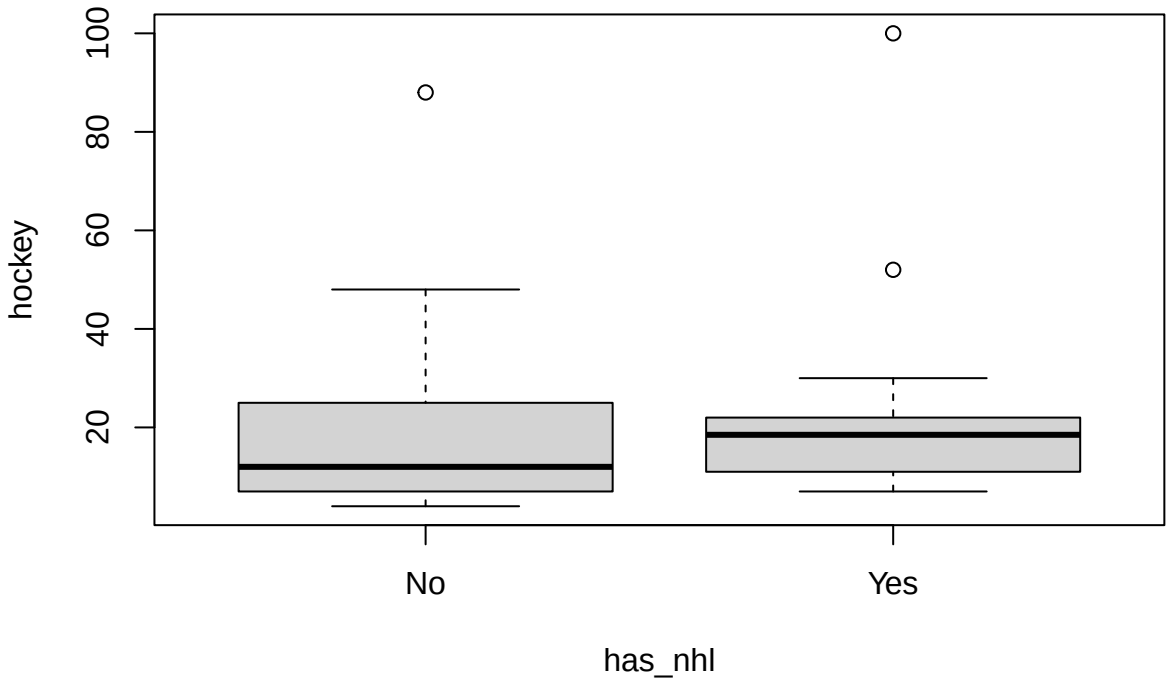



```
# Who are the outliers on "hockey"?
df |>
  filter(hockey > 45) |>
  select(state, hockey) |>
  arrange(desc(hockey))
```

```
## # A tibble: 5 x 2
##   state         hockey
##   <fct>          <dbl>
## 1 Minnesota        100
## 2 North Dakota      88
## 3 Massachusetts     52
## 4 Vermont           48
## 5 New Hampshire     46
```

BOXPLOTS BY GROUP

```
# Boxplots by group using plot()
df |>
  select(has_nhl, hockey) |>
  #filter_all(all_vars(is.finite(.))) |>
  plot()
```




```
#df %>%
#  mutate(has_nhl = factor(has_nhl)) %>%
#  ggplot(aes(x = hockey, fill = has_nhl)) +
#  geom_density(alpha = 0.5) +
#  labs(x = "Hockey", y = "Density", fill = "NHL Player")
```

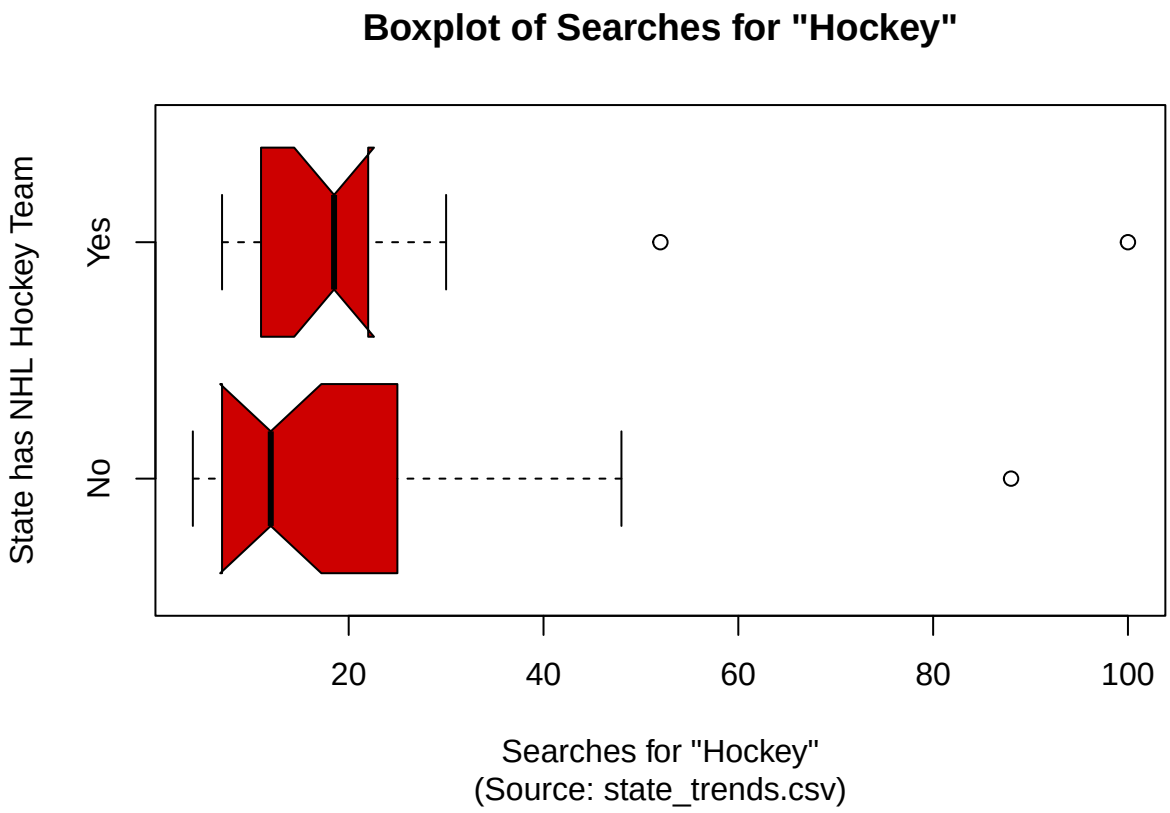
```
# Who is the outlier on "No"?
df |>
  filter(has_nhl == "No") |>
  filter(hockey > 80) |>
  select(state, hockey)
```

```
## # A tibble: 1 x 2
##   state        hockey
##   <fct>         <dbl>
## 1 North Dakota     88
```

```
# Boxplots by group using plot()
df |>
  select(has_nhl, hockey) |>
  plot(
    horizontal = T,  # Horizontal
    notch  = T,      # Confidence interval for median
    main   = "Boxplot of Searches for \"Hockey\"",
    sub    = "(Source: state_trends.csv)",
```

```
    xlab   = "Searches for \"Hockey\"",
    ylab   = "State has NHL Hockey Team",
    col    = "#CD0000"  # red3
  )
```

```
## Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, : some
## notches went outside hinges ('box'): maybe set notch=FALSE
```



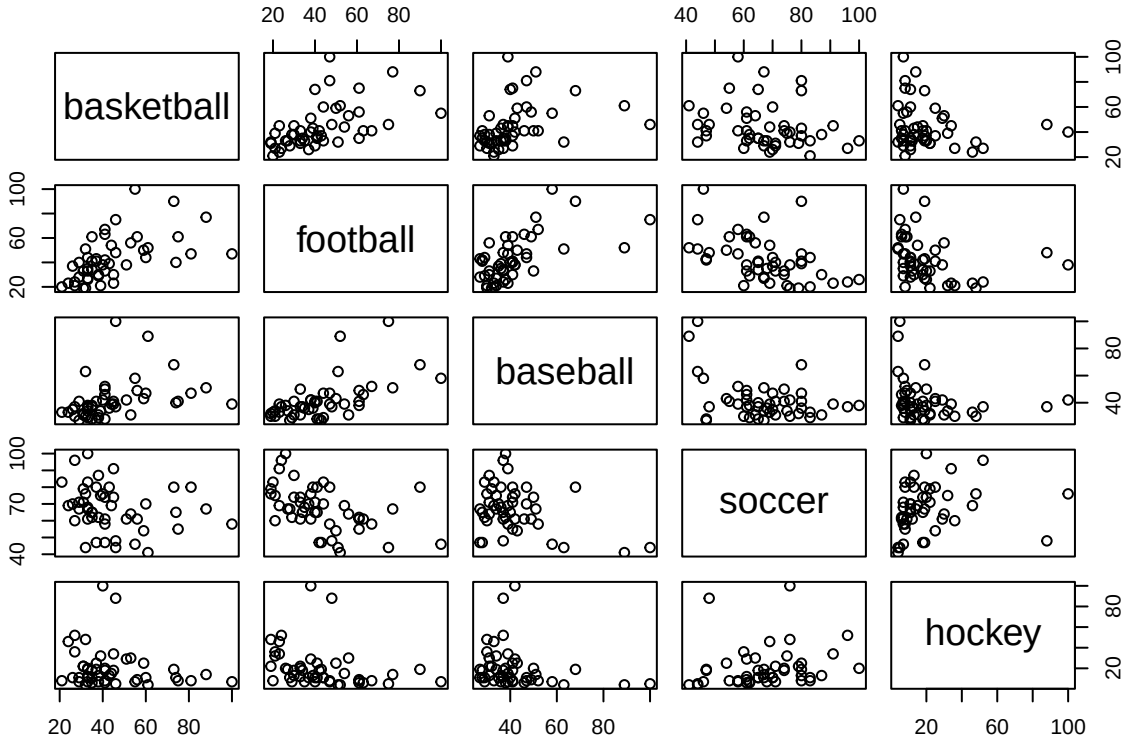
SCATTERPLOTS

```
df <- read_csv("../data/state_trends.csv") |>
  select(basketball:hockey) |>
  glimpse()
```

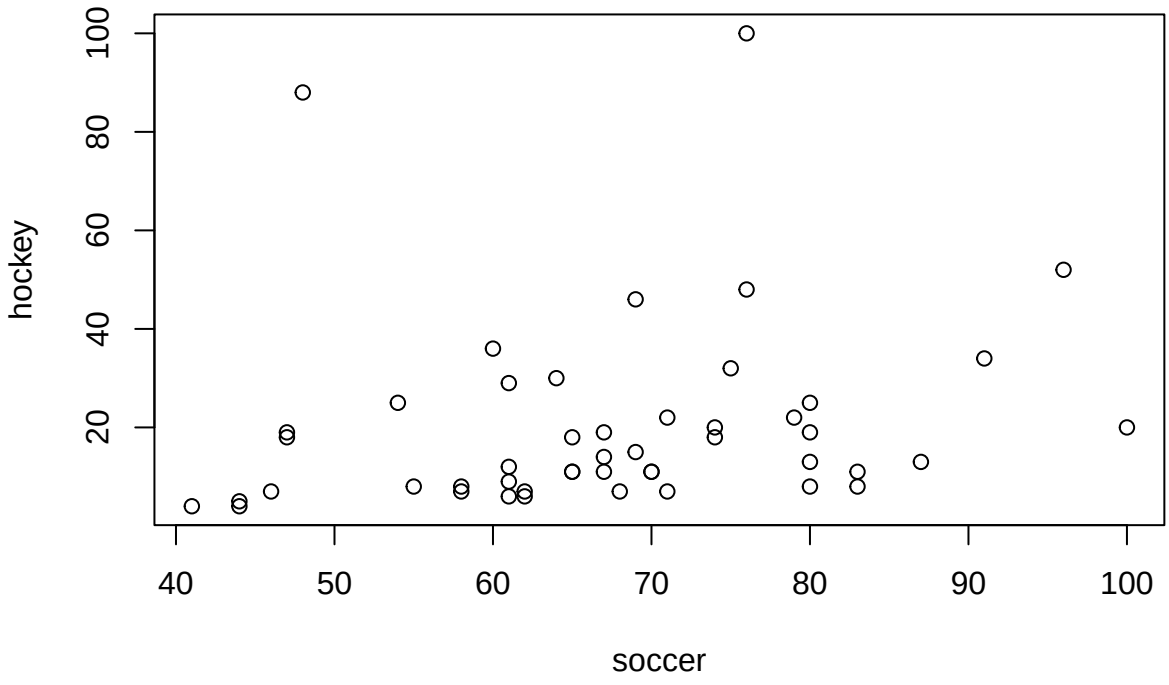
```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## Rows: 48
## Columns: 5
## $ basketball <dbl> 55, 33, 61, 21, 31, 45, 35, 26, 35, 35, 45, 74, 88, 81, 100~
## $ football   <dbl> 100, 33, 52, 20, 33, 23, 34, 37, 61, 41, 30, 40, 77, 47, 47~
## $ baseball   <dbl> 58, 37, 89, 33, 31, 39, 37, 35, 38, 28, 41, 40, 51, 47, 39,~
## $ soccer     <dbl> 46, 61, 41, 83, 71, 91, 65, 70, 62, 65, 74, 65, 67, 80, 58,~
## $ hockey     <dbl> 7, 12, 4, 8, 22, 34, 18, 11, 6, 11, 18, 11, 14, 8, 7, 4, 36~
```

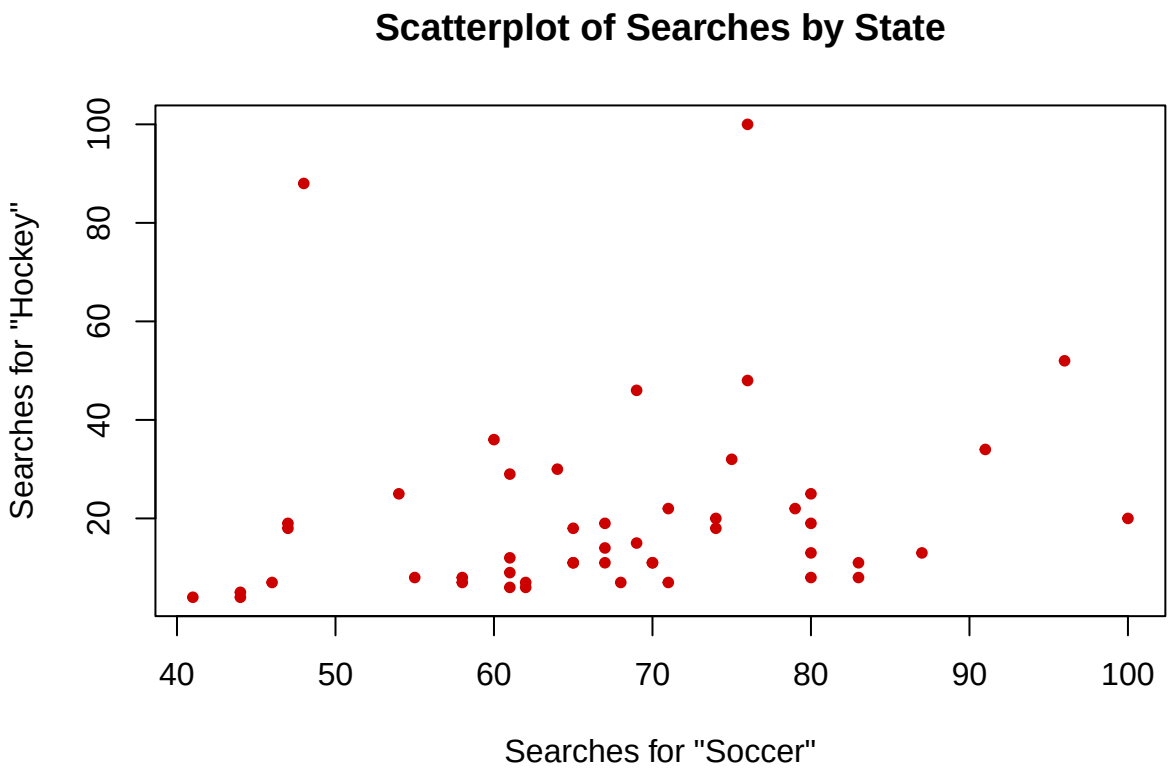
```
# Plot all associations
df |> plot()
```



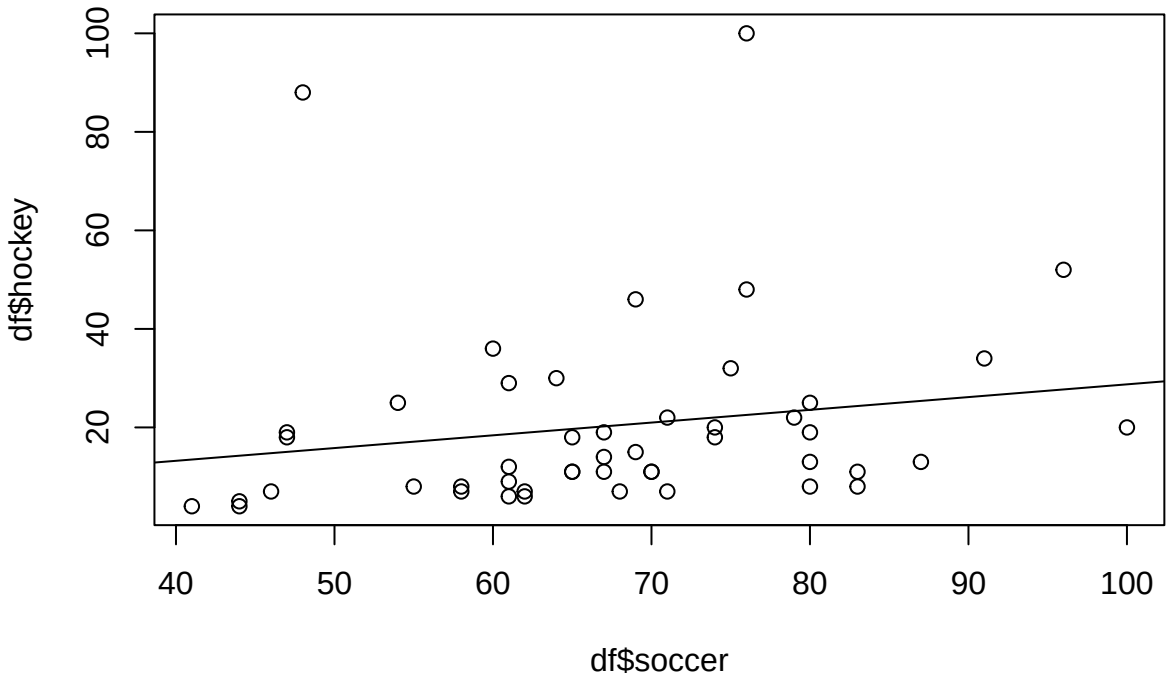

```
# Bivariate scatterplot with defaults
df |>
  select(soccer, hockey) |>
  plot()
```




```
# Bivariate scatterplot with options
df |>
  select(soccer, hockey) |>
  plot(
    main = "Scatterplot of Searches by State",
    xlab = "Searches for \"Soccer\"",
    ylab = "Searches for \"Hockey\"",
    col  = "red3",  # Color of points
    pch  = 20,      # "Plotting character" (small circle)
  )
```




```
# Add fit linear regression line (y ~ x)
plot(df$soccer, df$hockey)
lm(df$hockey ~ df$soccer) |>
  abline()
```



LINE CHART

```
# Load packages
library(datasets)   # Loads the built-in datasets
```

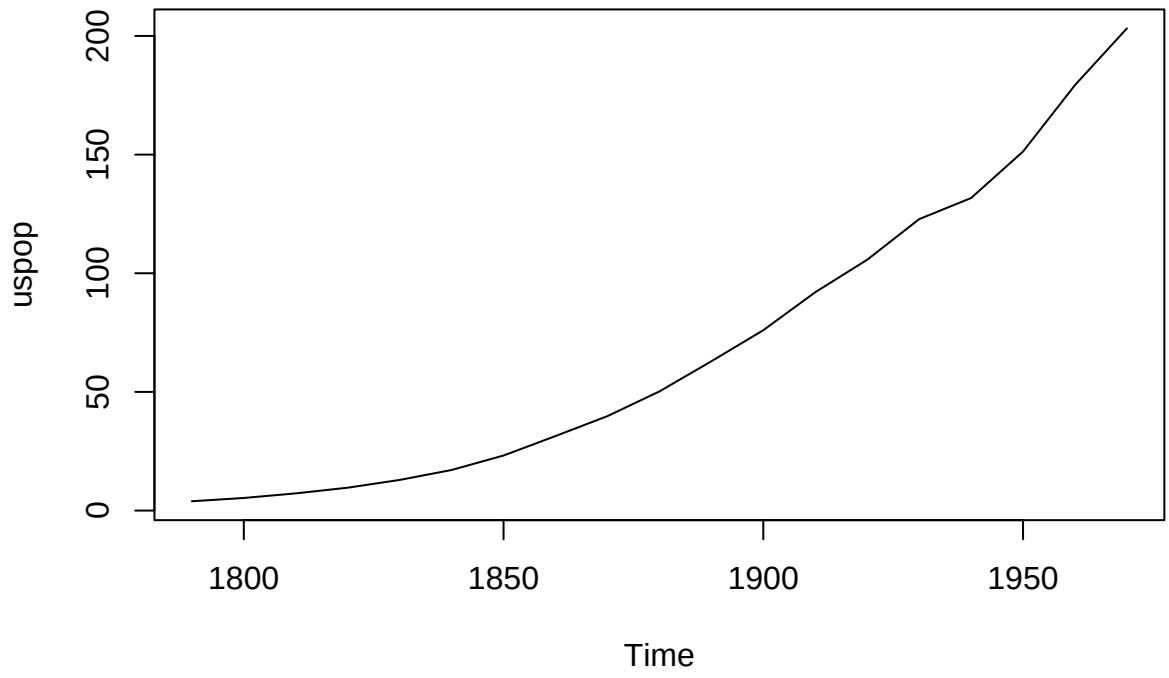
SINGLE TIME SERIES

uspop

#?uspop # Get info about data #uspop # Display data

#?ts # Get info about time-series objects

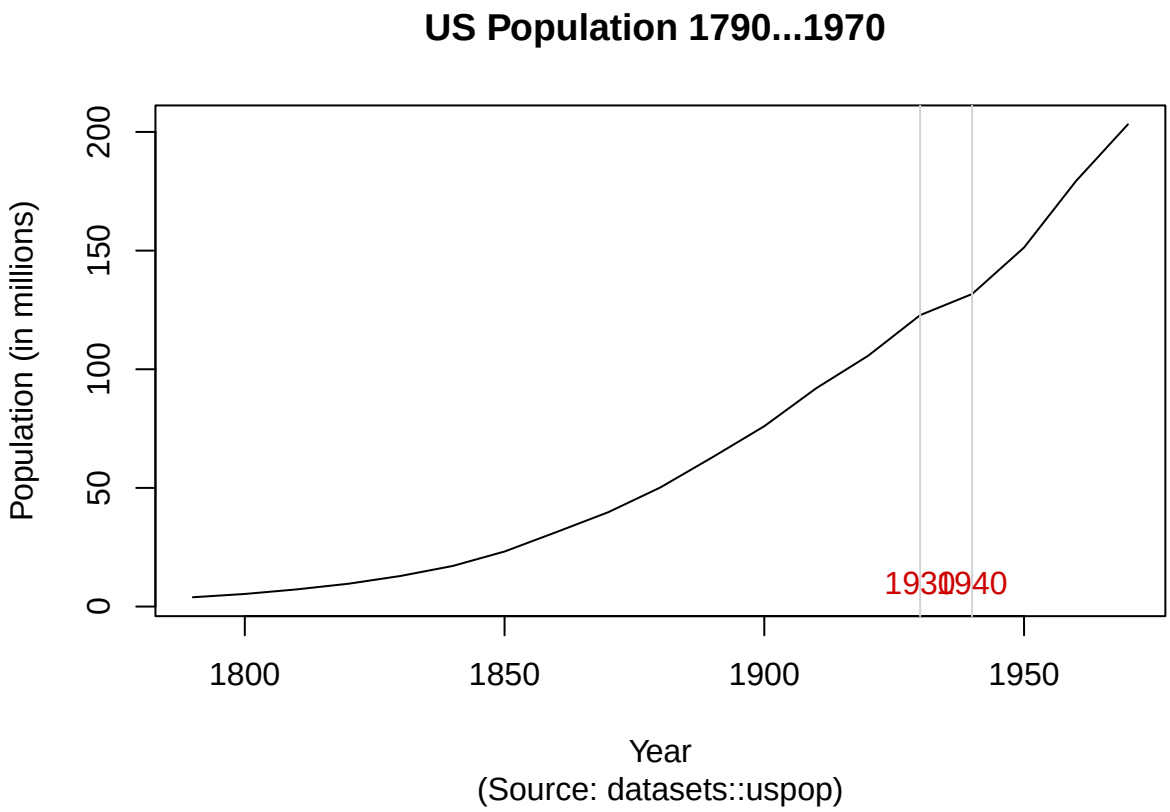
```
# Plot with default plot()
plot(uspop)
```



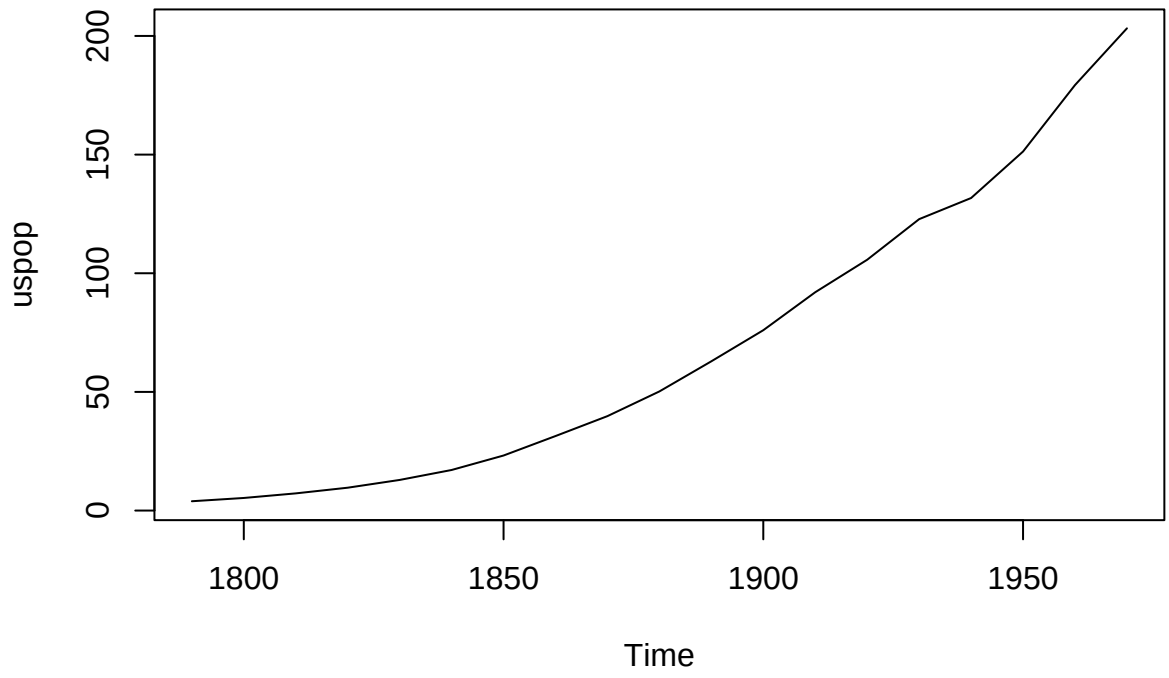

```
# Plot with options
uspop |>
  plot(
    main = "US Population 1790-1970 ",
    sub  = "(Source: datasets::uspop)",
    xlab = "Year",
    ylab = "Population (in millions)",
  )
```

```
## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): conversion failure
## on 'US Population 1790-1970 ' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): conversion failure
## on 'US Population 1790-1970 ' in 'mbcsToSbcs': dot substituted for <80>

## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): conversion failure
## on 'US Population 1790-1970 ' in 'mbcsToSbcs': dot substituted for <93>
```

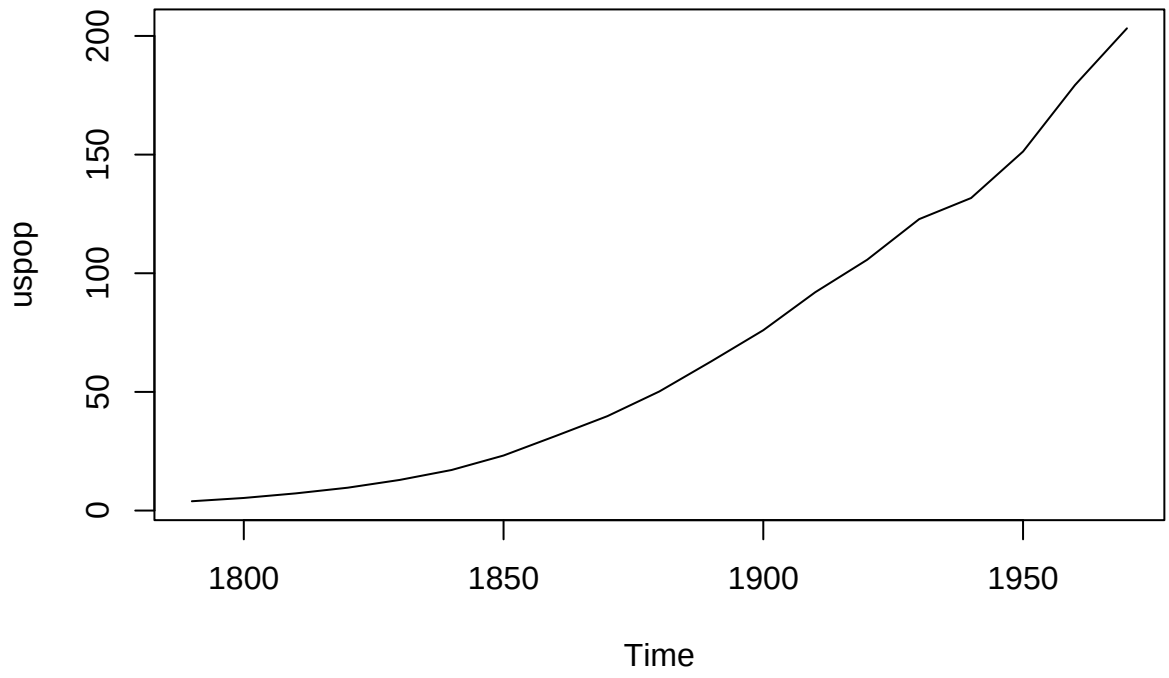
```
abline(v = 1930, col = "lightgray")
text(1930, 10, "1930", col = "red3")
abline(v = 1940, col = "lightgray")
text(1940, 10, "1940", col = "red3")
```




```
# Plot with ts.plot()
#?ts.plot
# Although this can be used for a single time series, plot
# is easier to use and is preferred.
ts.plot(uspop)
```



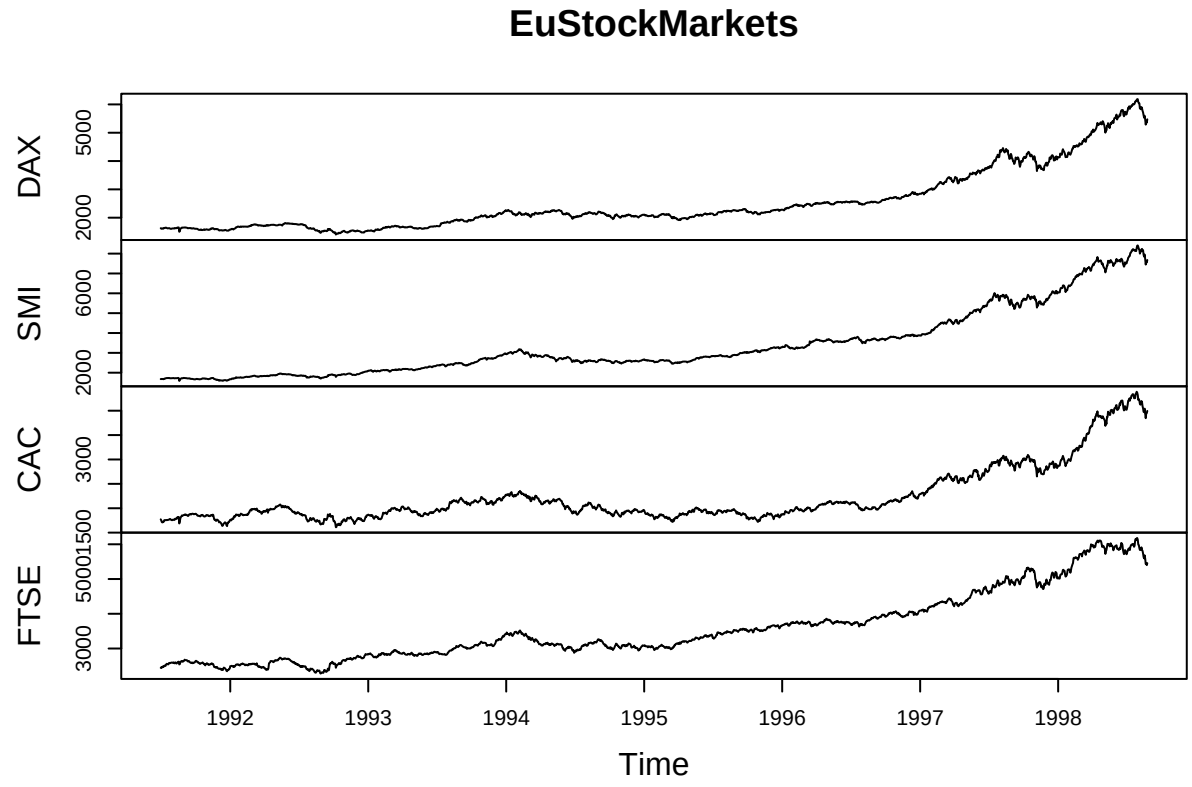

```
# Plot with plot.ts()
# More powerful alternative
?plot.ts
plot.ts(uspop)
```



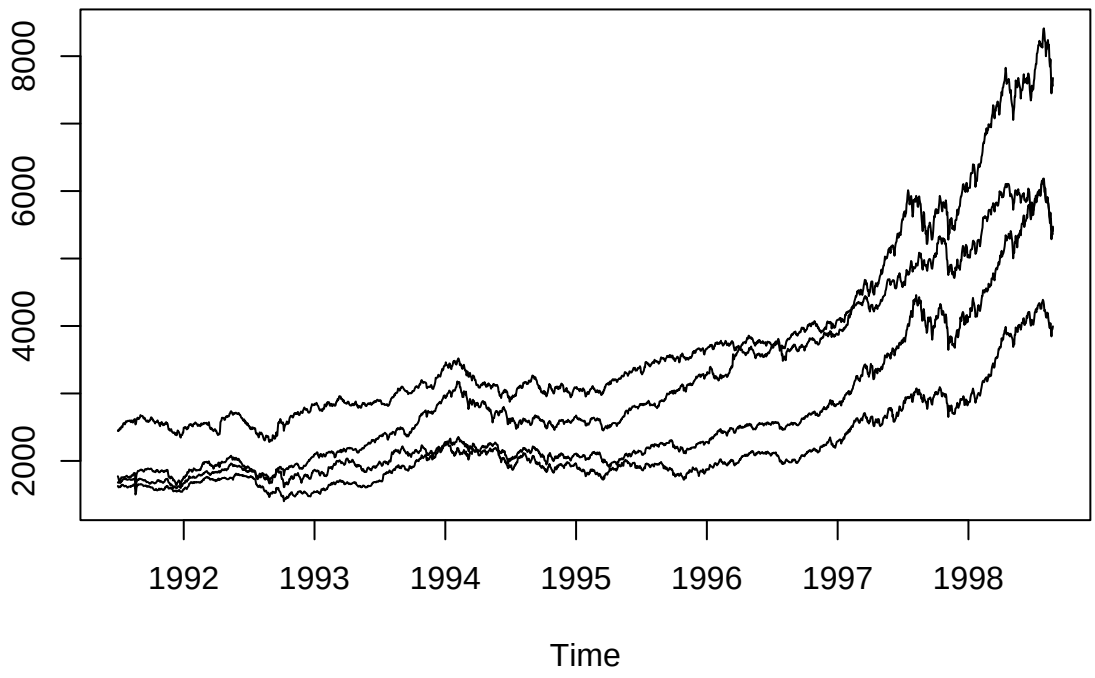
MULTIPLE TIME SERIES

```
# EuStockMarkets
# DAX (Germany), SMI (Switzerland), CAC (France), FTSE (UK)
#?EuStockMarkets
#EuStockMarkets
```

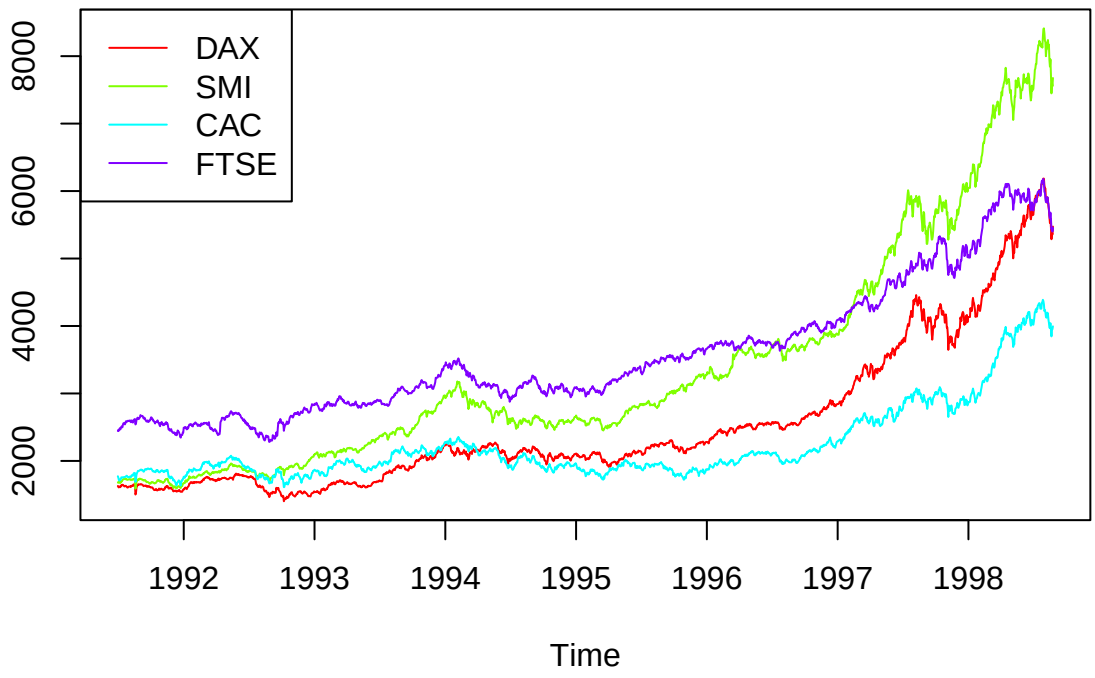
```
# Three different plot functions
plot(EuStockMarkets)     # Stacked windows
plot.ts(EuStockMarkets)  # Identical
```




```
ts.plot(EuStockMarkets)  # One window
```




```
# Plot with options
ts.plot(
  EuStockMarkets,
  col = rainbow(4))  # Color lines
legend(              # Add legend
  "topleft",         # Position
  legend = colnames(EuStockMarkets),  # Names for legend
  col = rainbow(4),  # Colors for legend
  lty = 1            # Line type: solid
)
```



CLUSTER CHART

```
# Select state codes and search data
df <- read_csv("../data/state_trends.csv") |>
  select(state_code, artificial_intelligence:hockey)
```

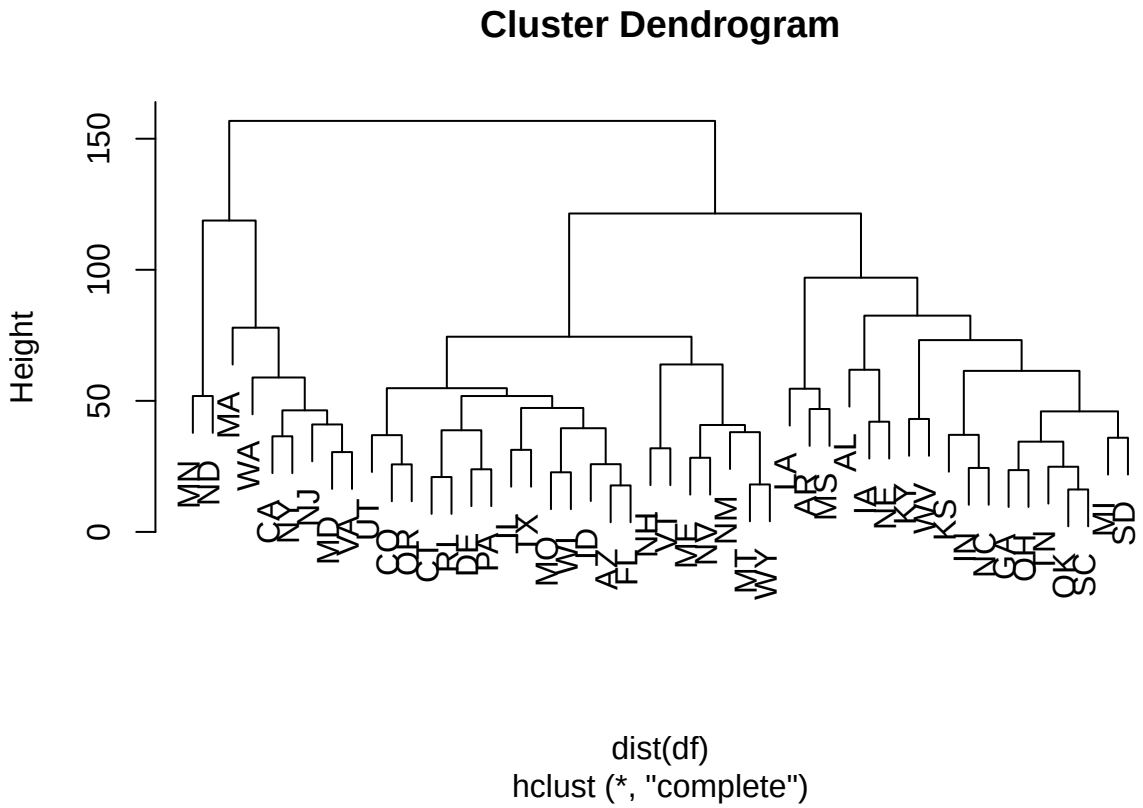
```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

ANALYZE DATA

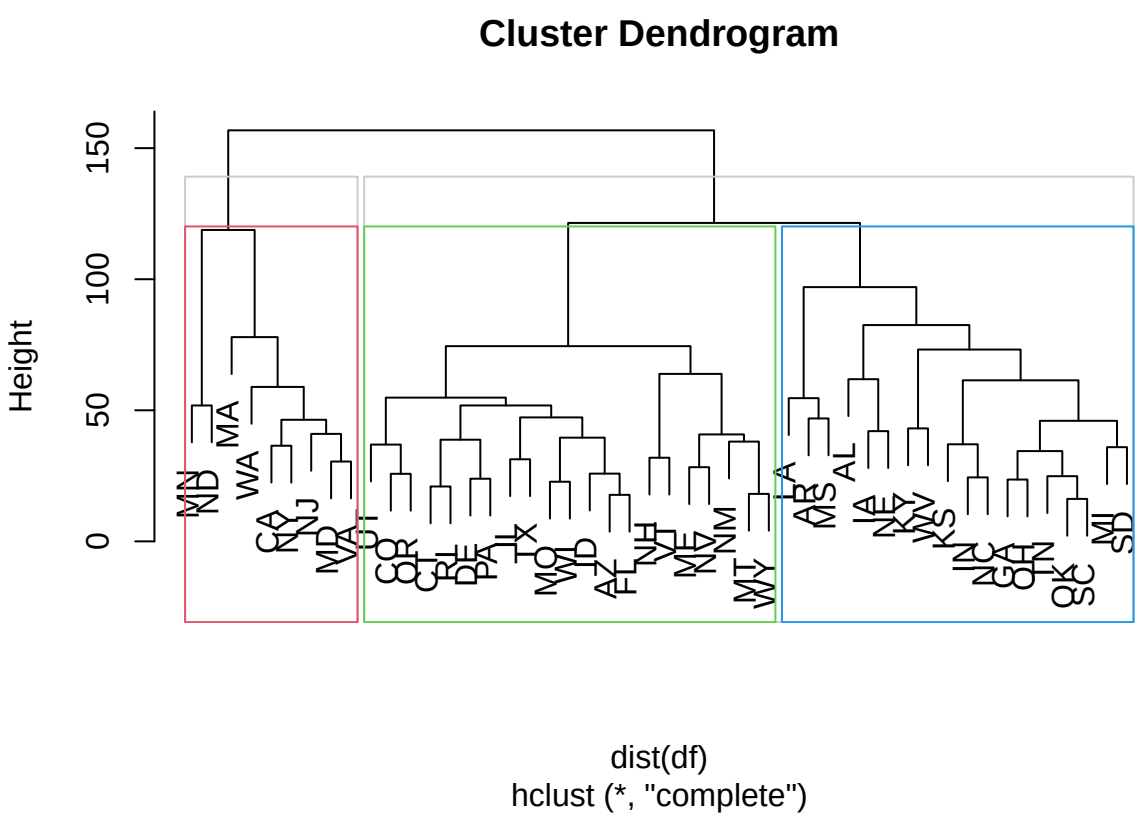
```
# Calculate clusters using hclust(), an agglomerative method
hc <- df |>  # Get data
  dist() |>  # Compute distance/dissimilarity matrix
  hclust()   # Compute hierarchical clusters
```

```
## Warning in dist(df): NAs introduced by coercion
```

```
# Plot dendrogram
hc |> plot(labels = df$state_code)
```




```
# Plot dendrogram
hc |> plot(labels = df$state_code)
# Draw boxes around clusters
hc |> rect.hclust(k = 2, border = "gray80")  # 2 boxes
hc |> rect.hclust(k = 3, border = 2:4)       # 3 boxes
```


Cluster Dendrogram
Height
0
50
100
150
MN
ND
MA
WA
CA
NJ
MD
VA
UT
CO
OR
CT
RI
DE
PA
TX
MO
WI
ID
AZ
FL
NH
VT
ME
NV
NM
MT
WY
IA
AR
MS
AL
LA
NE
KY
WV
KS
IN
NC
GA
OH
OK
SC
MI
SD
dist(df)
hclust (*, "complete")