Intermediate RStudio Training

December 13, 2017

GSA

D2D
DATA TO DECISIONS

Course Outline

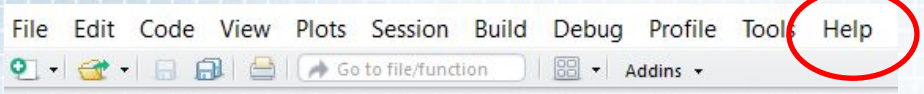
- Productivity features
- Essential data skills
- Factors
- Descriptive statistics
- Covariances and correlations

GSA

D2D
DATA TO DECISIONS

Productivity in RStudio

RStudio Cheat Sheet



Write Code

Navigate tabs

Open in new window

Save

Find and replace

Compile as notebook

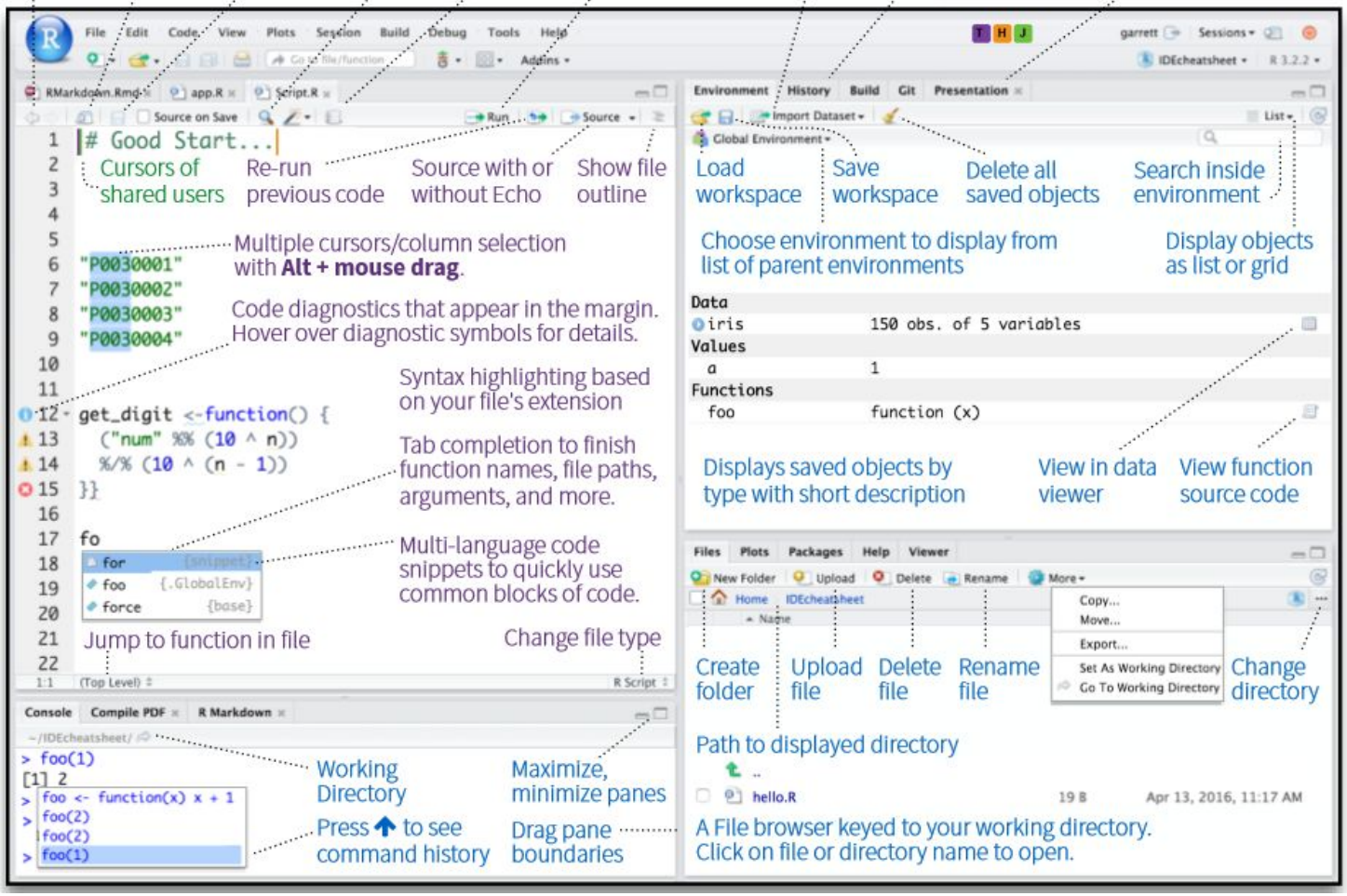
Run selected code

R Support

Import data with wizard

History of past commands to run/copy

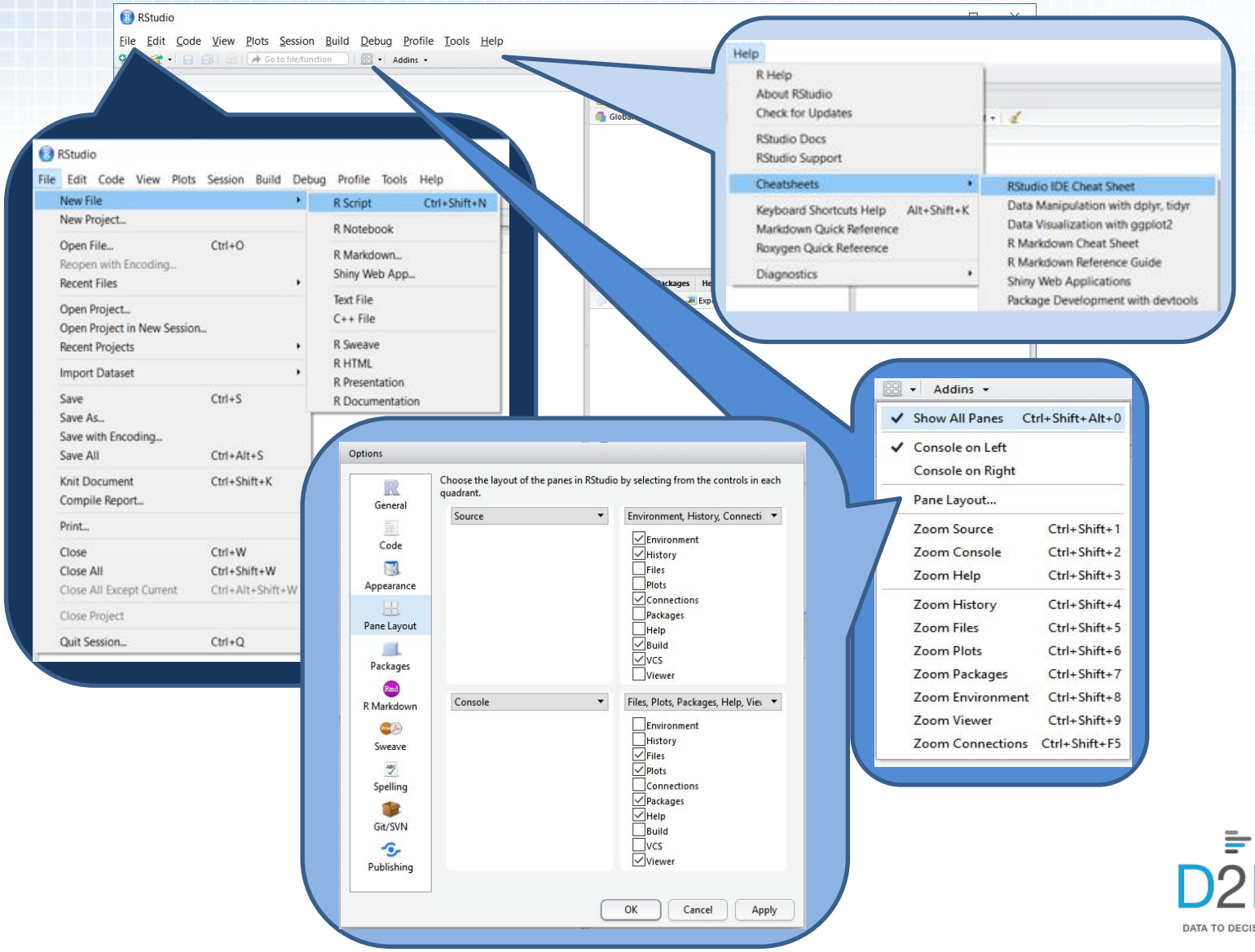
Display .RPres slideshows **File > New File > R Presentation**



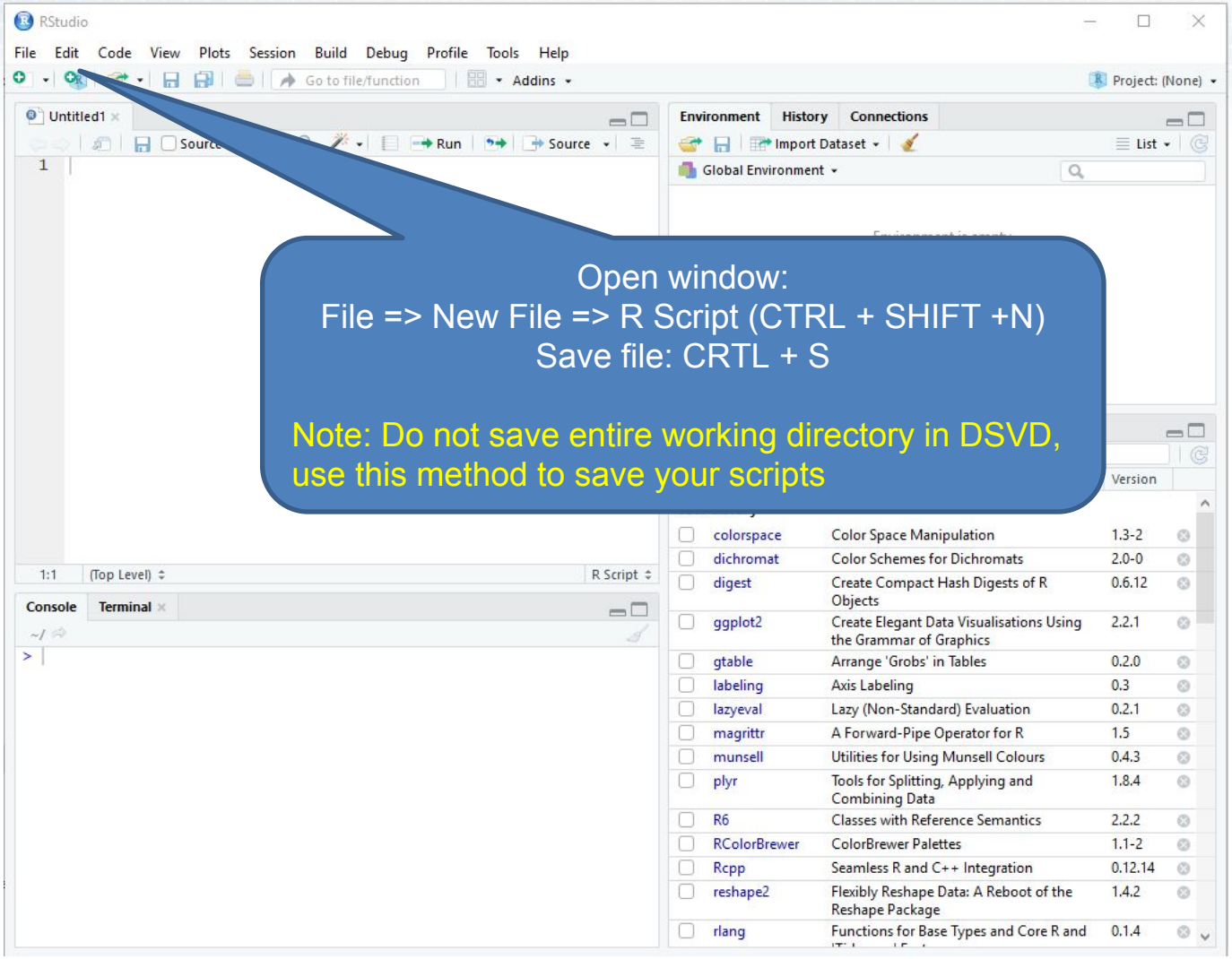
GSA

D2D DATA TO DECISIONS

Customizing RStudio



Scripting Window



GSA

D2D
DATA TO DECISIONS

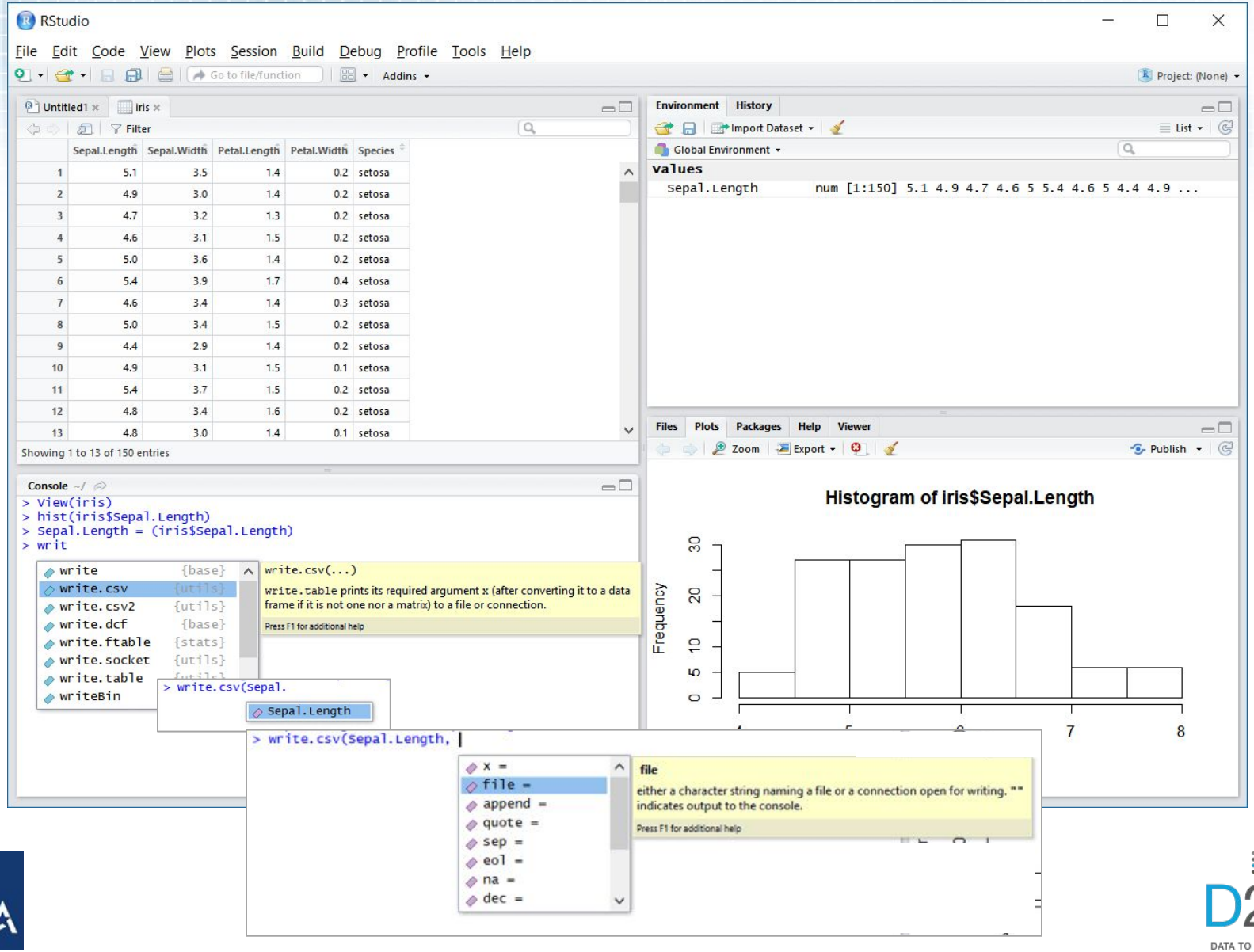
Auto-completion in RStudio

- **Tab** is a generic auto-complete function. If you start typing in the console or editor and hit the tab key, RStudio will suggest functions or file names; simply select the one you want and hit either tab or enter to accept it.
- **Control + the up arrow** (command + up arrow on a Mac) is a similar auto-complete tool. Start typing and hit that key combination, and it shows you a list of every command *you've* typed starting with those keys. Select the one you want and hit return. This works only in the interactive console, not in the code editor window.
- **Control + enter** (command + enter on a Mac) takes the current line of code in the editor, sends it to the console and executes it. If you select multiple lines of code in the editor and then hit ctrl/cmd + enter, all of them will run.

GSA

D2D
DATA TO DECISIONS

Exercise 1: Auto-completion



GSA

D2D
DATA TO DECISIONS

Essential Data Skills

GSA

D2D
DATA TO DECISIONS

Connect to MySQL in DSVD

- **Establish connection**

```
> library(DBI)
> library(RMySQL)
> mysqlconnection = dbConnect(MySQL(), user = 'd2dtraining_user',
password=                    , dbname='GSA_D2D_Training', host
='dc1dbeywj57czbx.c1q8kedtqajq.us-east-1.rds.amazonaws.com')
```

- **Read tables**

```
> dbListTables(mysqlconnection)
[1] "pseudo_facebook"          "reddit"
[3] "statesDataForIntermediateR"
```

- **Create R object**

```
> statesData = dbSendQuery(mysqlconnection, "select * from
statesDataForIntermediateR")
```

- **Convert object to data frame**

```
> dataFrame = fetch(statesData)
> print(dataFrame)s, or
> View(dataFrame)
```

Will be disabled after training
Replace with your own credentials / schema name

GSA

D2D
DATA TO DECISIONS

Read, Dim, Write

- **Read in data**

  > statesInfo = read.csv("C:\\Users\\mashk\\Desktop\\R Training\\statesDataForIntermediateR.csv")

- **Assess the data**

  > dim(statesInfo)

  [1] 50 12

  > class(statesInfo)

  [1] „data.frame“

- **Write data**

  > write.csv(statesInfo, file = „statesInfo.csv“)

- **Find your file**

  > getwd()

Replace “\”
with “/” or “\\”

GSA

D2D
DATA TO DECISIONS

Factors and Factor Levels

- **str() – compactly display the internal structure of an R object**
  - ➢ **Factor – a (categorical) variable with a discrete set of values**
  - ➢ **Level - a value a qualitative (categorical) variable can take**

```
> str(iris)
'data.frame':        150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 .
```

Subset Data Frame

- **Select rows with [ ]**

  > iris[1:5,]

- **Select columns with [ ]**

  iris[, c("Sepal.Length", "Sepal.Width")]

- **subset() function**

  StatesRegion1 = subset(statesInfo, state.region == 1)

  diamonsSelected = subset(diamonds, cut == "Ideal" & color == "E")

  dsPrice2 = subset(diamonds, cut == "Ideal" & color == "E", c("price", "clarity"))

  irisPetalData = subset(iris, select = -c(1,2))

- Can specify Column after “,”
- Use c() for multiple Columns
- All Columns if empty

- Drop Columns by number

GSA

D2D DATA TO DECISIONS

Descriptive Statistics

- **R provides a wide range of functions for obtaining summary statistics.**
- **Built-in functions sd(), mean(), median(), max/min(), e.g.:**
  - ➢ mean(x, trim = 0, na.rm = FALSE)

na – not available
rm - remove

```
> mean(iris$Sepal.Length)
[1] 5.843333
> mean(iris$Sepal.Length, trim = .05)
[1] 5.820588
```

- **summary()**
  - ➢ summary(StatesRegion1)
  - ➢ **Quartiles:**
    - **1stQu** or lower quartile, is the value that cuts off the first 25% of the data when it is sorted in ascending order. i.e. 70.55 life.exp means that 25% live less than 70.55 years.
    - second quartile, or **Median**, is the value that cuts off the first 50%
    - **3rdQu** , or upper quartile, is the value that cuts off the first 75%

```
   life.exp        murder       highSchoolGrad      frost            area
Min.   :70.39   Min.   : 2.400   Min.   :46.40   Min.   : 82.0   Min.   : 1049
1st Qu.:70.55   1st Qu.: 3.100   1st Qu.:52.50   1st Qu.:115.0   1st Qu.: 7521
Median :71.23   Median : 3.300   Median :54.70   Median :127.0   Median : 9027
Mean   :71.26   Mean   : 4.722   Mean   :53.97   Mean   :132.8   Mean   :18141
3rd Qu.:71.83   3rd Qu.: 5.500   3rd Qu.:57.10   3rd Qu.:161.0   3rd Qu.:30920
Max.   :72.48   Max.   :10.900   Max.   :58.50   Max.   :174.0   Max.   :47831
```

GSA

D2D DATA TO DECISIONS

Covariances and Correlations

- **Same syntax for both**
- **Correlation between x and y**

  cor(x,y), cov(x,y)

If x is a matrix, then y is NULL and syntax is cor(x)

- **Optional parameters**
  - ➢ **Use**

    A character string giving a method for computing in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs“

    - **Correlation with missing data**

      cor(x,y, use = “complete”)

  - ➢ **Method**

A character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), "kendall", or "spearman":

```
> cor(iris$Sepal.Length, iris$Petal.Length)
[1] 0.8717538
> cor(iris$Sepal.Length, iris$Petal.Length, method = "kendall")
[1] 0.7185159
> cor(iris$Sepal.Length, iris$Petal.Length, method = "spearman")
[1] 0.8818981
> cov(iris$Sepal.Length, iris$Petal.Length, method = "spearman")
[1] 1661.304
```

GSA

D2D DATA TO DECISIONS

Correlation Test

- **Can be based on Pearson, Kendall or Spearman**
- **Returns correlation coefficient and p-value (significance level)**

```
> cor.test(iris$Sepal.Length, iris$Petal.Length)

Pearson's product-moment correlation

data:  iris$Sepal.Length and iris$Petal.Length
t = 21.646, df = 148, p-value < 0.00000000000000022
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8270363 0.9055080
sample estimates:
      cor
0.8717538
```

GSA

D2D
DATA TO DECISIONS

Pearson Vs Kendal and Spearman

- Most commonly used is Pearson method
- Pearson correlation measures linear dependence between two normally distributed variables when change in one variable causes proportional change in another
- Kendal and Pearson are rank-based correlation coefficients, i.e. the data point are first ranked and the correlation of the ranks is measured
- Kendal and Pearson can be used for ordinal data, e.g. correlation between test scores and months of training

GSA

D2D
DATA TO DECISIONS

ggplot2

- "ggplot2 is the most elegant and aesthetically pleasing graphics framework available in R. It has a nicely planned structure to it."

  http://r-statistics.co/ggplot2-Tutorial-With-R.html

- ggplot2 works with data frames and not individual vectors
- You can keep enhancing the plot by adding more layers (and themes) to an existing plot created using the ggplot() function

GSA

D2D
DATA TO DECISIONS

ggplot2 Cont.

- **Setup**

```
options(scipen=999)  # turn off scientific notation like 1e+06
library(ggplot2)
data("midwest", package = "ggplot2")
```

- **Initiate**

```
ggplot(midwest, aes(x=area, y=poptotal))
```

- **Scatter Plot**

```
ggplot(midwest, aes(x=area, y=poptotal)) + geom_point()
```

- **Smoothing layer**

```
g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() +
geom_smooth(method="lm")
plot(g)
```

- **Adjust axis by deleting out of range points**

```
g + xlim(c(0, 0.1)) + ylim(c(0, 1000000))
```

- **Change labels**

```
g1 <- g + coord_cartesian(xlim=c(0,0.1), ylim=c(0, 1000000))
g1 + labs(title="Area Vs Population", subtitle="From midwest dataset",
y="Population", x="Area", caption="Midwest Demographics")
```

GSA

D2D
DATA TO DECISIONS

ggplot2 Cont. 2

- **Change color and size of points**

```
ggplot(midwest, aes(x=area, y=poptotal)) +
  geom_point(col="steelblue", size=3) +   # Set static color and size for
points
  geom_smooth(method="lm", col="firebrick") +  # change the color of line
  coord_cartesian(xlim=c(0, 0.1), ylim=c(0, 1000000)) +
  labs(title="Area Vs Population", subtitle="From midwest dataset",
y="Population", x="Area", caption="Midwest Demographics")
```

- **Change the Color To Reflect Categories in Another Column**

```
gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
  geom_point(aes(col=state), size=3) +
  geom_smooth(method="lm", col="firebrick", size=2) +
  coord_cartesian(xlim=c(0, 0.1), ylim=c(0, 1000000)) +
  labs(title="Area Vs Population", subtitle="From midwest dataset",
y="Population", x="Area", caption="Midwest Demographics")
plot(gg)
```

GSA

D2D
DATA TO DECISIONS

Q & A