



Data Science with R

Statistics – ITS

Regita Putri Permata

(Master Student at Statistics Department – ITS)

Meet 1 : Introduction R

Outline

- Introduction R and R Studio
 - Basic Calculation
 - If Statement
 - Looping Statement
 - Type of data
 - Read data and write data
- Visualization Data and Summary Data
 - Pie Chart, Bar Chart
 - Histogram, Box Plot
 - Scatterplot, Matrix Scatterplot
 - Summary data/Descriptive statistics (Mean, Median, Q1, Q3, St. Dev)
 - Make a simple analysis
- Let's Practice!

Introduction R and R Studio



R is a language and environment for statistical computing and graphics. Available at <https://cran.r-project.org/>



RStudio allows the user to run R in a more user friendly environment. It is open source (i.e. free) and available at <http://www.rstudio.com/>

Introduction R and R Studio

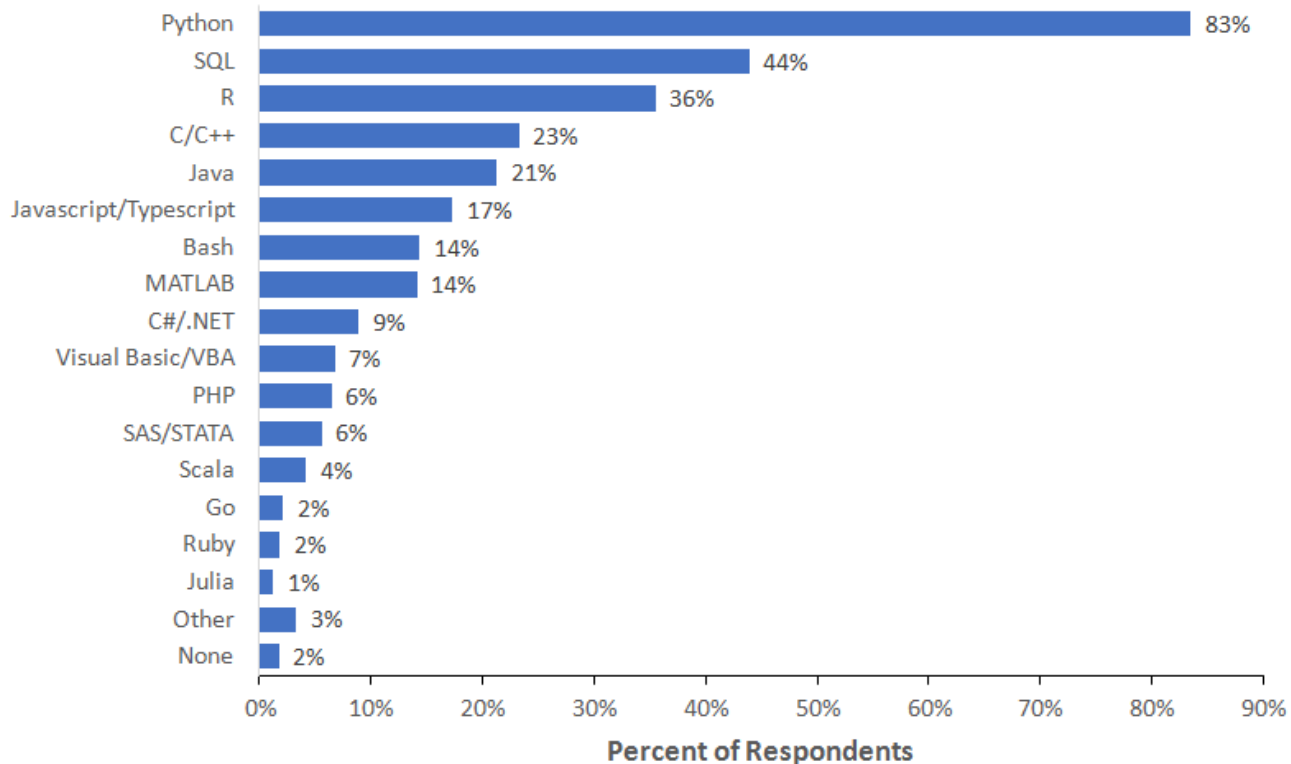
Why use R (R Studio)?

- **Data analysis software:** R is a data analysis software. It is used by data scientists for statistical analysis, predictive modeling and visualization.
- **Statistical analysis environment:** R provides a complete environment for statistical analysis. It is easy to implement statistical methods in R. Most of the new research in statistical analysis and modeling is done using R. So, the new techniques are first available only in R.
- **Open source:** R is open source technology, so it is very easy to integrate with other applications.
- **Community support:** R has the community support of leading statisticians, data scientists from different parts of the world and is growing rapidly.
- **It's free:** R uses free packages to analyze data

Introduction to R

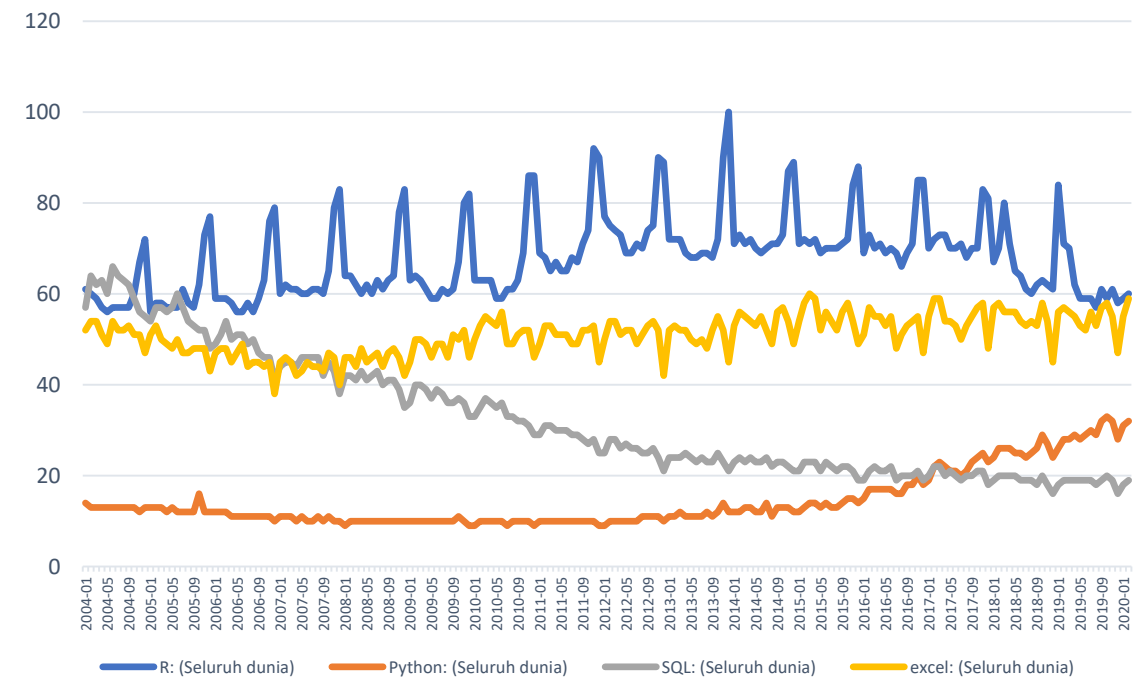


What programming language do you use on a regular basis?



Note: Data are from the 2018 Kaggle Machine Learning and Data Science Survey. You can learn more about the study here: <http://www.kaggle.com/kaggle/kaggle-survey-2018>. A total of 18827 respondents answered the question.

Software for Data Science 2004-2020



Introduction to R



Aritmathic function in R/R Studio and Variable Assignment

In its most basic form, R can be used as a simple calculator. Consider the following arithmetic operators:

- Addition: +
- Subtraction: -
- Multiplication: *
- Division: /
- Exponentiation: ^
- Modulo: %%

```
> 4+3-2
[1] 5
(6+18)/6
[1] 4
> 2**3
[1] 8
2^3
[1] 8
> 2^(2*3)
[1] 64
```

```
x=3; x
[1] 3
> y=4 ; y
[1] 4
> z=x+y;z
[1] 7
```

names are case sensitive

- pi is a constant, but still can be used as variable name.
- print(x) prints content of x

Introduction R and R Studio

If Statement

```
# simple if
x<-1
if (x==2){ print ("x=2") }
x
[1] 1
# if - else
x<-1
if (x>=0){x="A"} else {x="B"}
x
[1] "A"
```

Logical Function	Meaning
<	smaller
<=	smaller or equal
>	bigger
>=	bigger or equal
!=	unequal
==	logical equal
!	logical NOT (unary)
&	logical AND (vector)
	logical OR (vector)
&&	logical AND (no vector)
	logical OR (no vector)

Introduction R and R Studio

Looping Statement (for)

```
for (i in 1:4) { print (i) }
```

```
[1] 1
[1] 2
[1] 3
[1] 4
```

```
for (i in letters [1:4]) { print (i) }
```

```
[1] "a"
[1] "b"
[1] "c"
[1] "d"
```

```
a<-numeric (20) # generate empty a of length 20
```

```
for (i in 1:20) { a[i]=i } # fill a with 1:20
```

```
a
```

```
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```


Introduction to R



Mathematical Function

Function	Meaning
<code>log(x)</code>	log to base e of x
<code>exp(x)</code>	antilog of x ($=2.7818x$)
<code>log(x,n)</code>	log to base n of x
<code>log10(x)</code>	log to base 10 of x
<code>sqrt(x)</code>	square root of x
<code>factorial(x)</code>	$x!$
<code>choose(n,x)</code>	binomial coefficients $n!/(x! (n - x)!)$
<code>gamma(x)</code>	$\Gamma.x.(x - 1)!$ for integer x
<code>lgamma(x)</code>	natural log of gamma(x)
<code>floor(x)</code>	greatest integer $< x$

Introduction to R



Mathematical Function

Function	Meaning
<code>ceiling(x)</code>	smallest integer $> x$
<code>trunc(x)</code>	closest integer to x between x and 0: <code>trunc(1.5) = 1</code> , <code>trunc(-1.5) = -1</code>
<code>trunc</code>	is like floor for positive values and like
<code>ceiling</code>	for negative values
<code>round(x, digits=0)</code>	round the value of x to an integer
<code>signif(x, digits=6)</code>	give x to six digits in scientific notation
<code>runif(n)</code>	generates n random numbers between 0 and 1 from a uniform distribution
<code>cos(x)</code>	cosine of x in radians
<code>sin(x)</code>	sine of x in radians
<code>tan(x)</code>	tangent of x in radians
<code>acos(x)</code> , <code>asin(x)</code> , <code>atan(x)</code>	inverse trigonometric transformations of real or complex numbers.
<code>acosh(x)</code> , <code>asinh(x)</code> , <code>atanh(x)</code>	inverse hyperbolic trigonometric transformations on real or complex numbers
<code>abs(x)</code>	the absolute value of x , ignoring the minus sign if there is one

Introduction to R

Basic data type in R

Main Structures

Vector array with 1 dimension in size m (1 data type)

Matrix array with 2 dimension in size $m \times n$ (1 data type)

Dataframe like matrix, but it contain more than 1 data type

Data Type

character vector of strings

numeric vector of real numbers

integer vector of signed integer

logical vector of boolean (TRUE or FALSE)

complex vector of complex numbers

list vector of R objects

factor sets of labelled observations, pre-defined set of labels

NA not available, missing value

Introduction to R



Vector in R

```
a=1:3
> b=2:4
> c(a,b)
[1] 1 2 3 2 3 4
> c(1,a) [1] 1 1 2 3
> array(1,3) [1] 1 1 1
> seq(1,5) [1] 1 2 3 4 5
> seq(from=1,to=3,length.out = 4) #desired length of the sequence
[1] 1.000000 1.666667 2.333333 3.000000
> AA <- letters [1:3] ;AA
[1] "a" "b" "c" > K <- c(3,2,1,3,2)
> length(K)
[1] 5
> K[2]
[1] 2
> K[1:3]
[1] 3 2 1
> K[-1]
[1] 2 1 3 2
```

Introduction to R



Type of data

Vector Function

Operation	Meaning
<code>max(x)</code>	maximum value in x
<code>min(x)</code>	minimum value in x
<code>sum(x)</code>	total of all the values in x
<code>mean(x)</code>	arithmetic average of the values in x
<code>median(x)</code>	median value in x
<code>range(x)</code>	vector of <code>min(x)</code> and <code>max(x)</code>
<code>var(x)</code>	sample variance of x, with degrees of freedom= <code>length(x) - 1</code>
<code>cor(x,y)</code>	correlation between vectors x and y
<code>sort(x)</code>	a sorted version of x
<code>rank(x)</code>	vector of the ranks of the values in x

Introduction R and R Studio

Vector Function

Operation	Meaning
<code>order(x)</code>	an integer vector containing the permutation to sort x into ascending order
<code>quantile(x)</code>	vector containing the minimum, lower quartile, median, upper quartile, and maximum of x
<code>cumsum(x)</code>	vector containing the sum of all of the elements up to that point
<code>cumprod(x)</code>	vector containing the product of all of the elements up to that point
<code>cummax(x)</code>	vector of non-decreasing numbers which are the cumulative maxima of the values in x up to that point.
<code>cummin(x)</code>	vector of non-increasing numbers which are the cumulative minima of the values in x up to that point.
<code>pmax(x,y,z)</code>	vector, of length equal to the longest of x, y, or z containing the maximum of x, y or z for the ith position in each
<code>pmin(x,y,z)</code>	vector, of length equal to the longest of x, y, or z containing the minimum of x, y or z for the ith position in each

Introduction to R



Matrix in R

```
> matriks.1 = matrix(c(1,2,3,4,5,6),nrow=2,ncol=3);matriks.1
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> matriks.2 = matrix(1:6,nrow=2,ncol=3) ; matriks.2
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> data=c(6.4,8.8,7.5,5.3,7.6,9.5) ; data
[1] 6.4 8.8 7.5 5.3 7.6 9.5
> matriks.a=matrix(data,nrow=3,ncol=2) ; matriks.a
      [,1] [,2]
[1,]  6.4  5.3
[2,]  8.8  7.6
[3,]  7.5  9.5
> dim(matriks.1)
[1] 2 3
```

Introduction to R



Operator Matrix in R

Operator	Note
*	Multiplication by element of matrix
%*%	Matrix multiplication
Solve	Inverse matrix
t	Transpose
crossprod	Crossproduct matrix $t(x) \%*\% x$

```
a=1:5
> a
[1] 1 2 3 4 5
> a*a [1] 1 4 9 16 25
> crossprod(a)
      [,1]
[1,] 55
> b=matrix(c(1:4),2)
> b
      [,1] [,2]
[1,] 1     3
[2,] 2     4
> b*b [1,] [,2]
[1,] 1 9
[2,] 4 16
> b%*%b
      [,1] [,2]
[1,] 7 15
[2,] 10 22
```

```
> solve(b)
      [,1] [,2]
[1,] -2   1.5
[2,] 1  -0.5
> c=c(3,5)
> d=cbind(b,c)
> d=cbind(b,c); d
c [1,] 1 3 3
  [2,] 2 4 5
> e=rbind(b,c); e
[,1] [,2]
     1 3
     2 4
c    3 5
```


Introduction to R



Example of vector and data frame

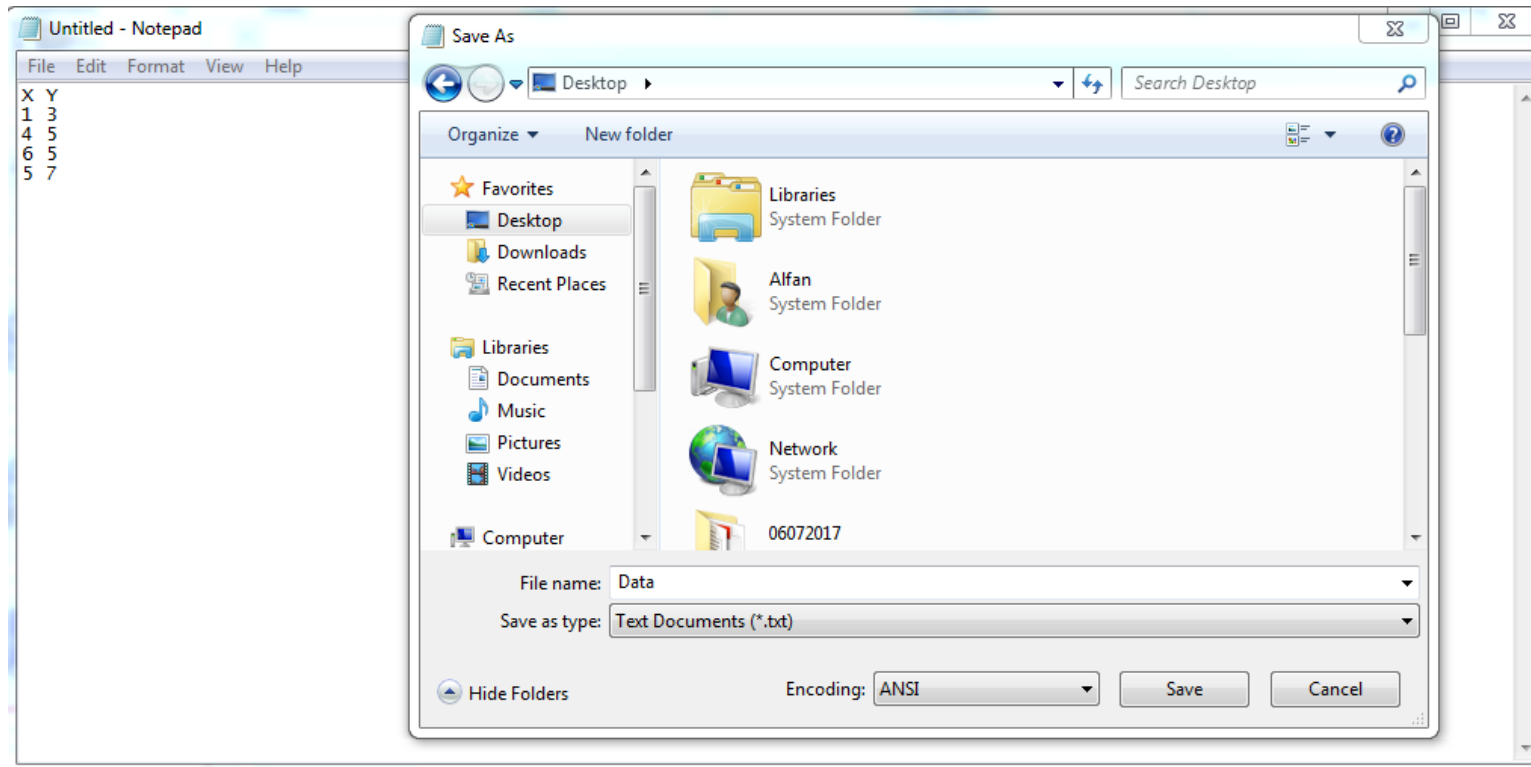
```
land=factor(c("Belgium","Denmark","France","GB","Ireland","Italy","Luxemburg",  
              "Holland","Portugal","Spain","USA","Japan","Deutschland"))  
x=c(2.8,1.2,2.1,1.6,1.5,4.6,3.6,2.1,6.5,4.6,3,1.3,4.2);  
y=c(9.4,10.4,10.8,10.5,18.4,11.1,2.6,8.8,5,21.5,6.7,2.5,5.6);  
data1=data.frame(land,x,y)  
colnames(data1)= c("countries","index","unemp")  
income= c(12,10,13,9,8,10,11,12,13.5,14,10,11,13);  
datanew = cbind(data1,income); datanew  
subset(datanew,income<13)  
subset(datanew$index,datanew$unemp>10) #mengambil data index sesuai ketentuan unemp>10
```

	countries	index	unemp		countries	index	unemp	income
1	Belgium	2.8	9.4	1	Belgium	2.8	9.4	12.0
2	Denmark	1.2	10.4	2	Denmark	1.2	10.4	10.0
3	France	2.1	10.8	3	France	2.1	10.8	13.0
4	GB	1.6	10.5	4	GB	1.6	10.5	9.0
5	Ireland	1.5	18.4	5	Ireland	1.5	18.4	8.0
6	Italy	4.6	11.1	6	Italy	4.6	11.1	10.0
7	Luxemburg	3.6	2.6	7	Luxemburg	3.6	2.6	11.0
8	Holland	2.1	8.8	8	Holland	2.1	8.8	12.0
9	Portugal	6.5	5.0	9	Portugal	6.5	5.0	13.5
10	Spain	4.6	21.5	10	Spain	4.6	21.5	14.0
11	USA	3.0	6.7	11	USA	3.0	6.7	10.0
12	Japan	1.3	2.5	12	Japan	1.3	2.5	11.0
13	Deutschland	4.2	5.6	13	Deutschland	4.2	5.6	13.0

Introduction to R



Read data and write data

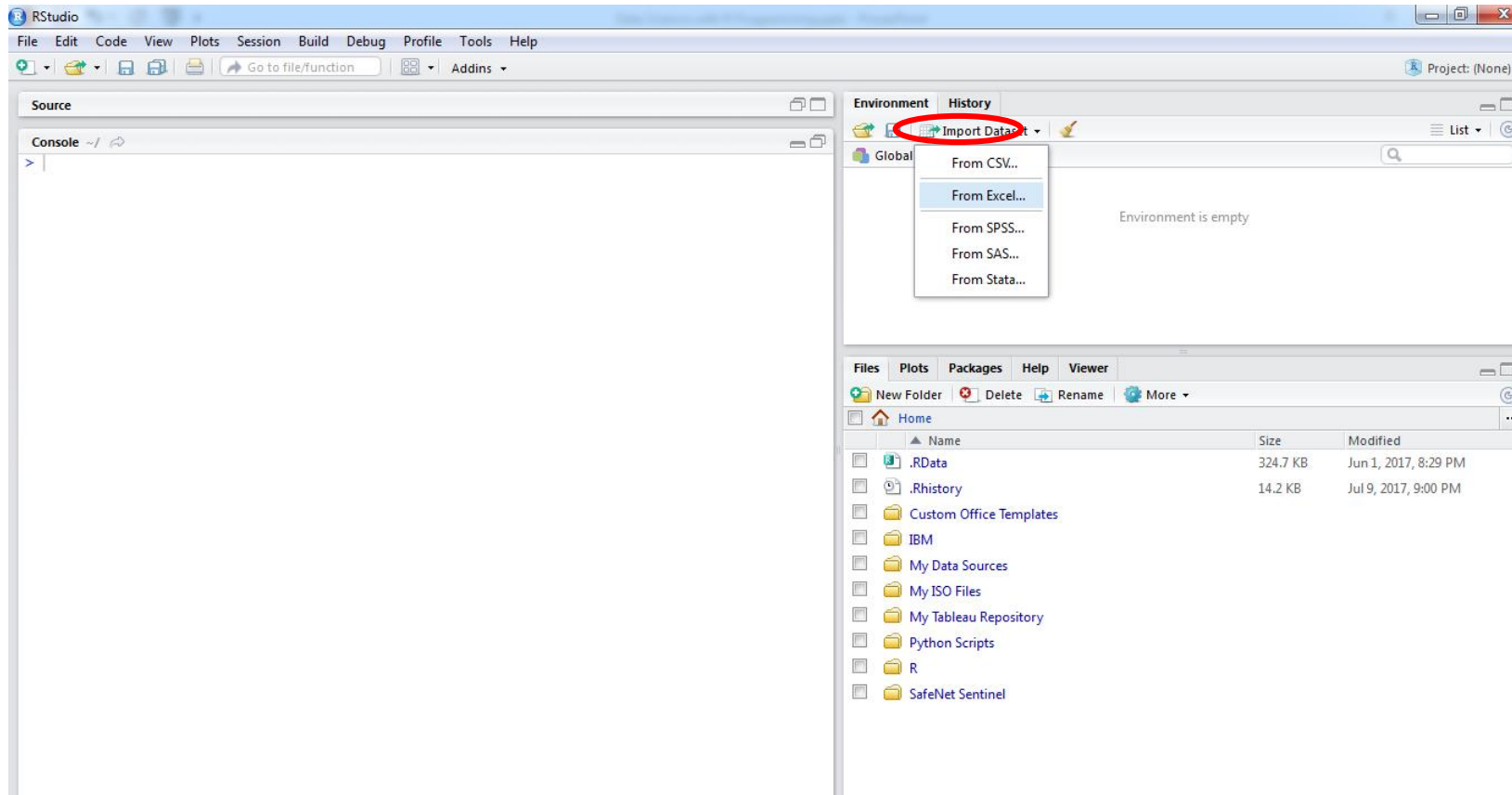


For example we create data in notepad

Introduction to R



Read data and write data



Click import dataset at R Studio,
choose from CSV

Introduction to R



Read data and write data

Import Text Data

File/Url:
C:/Users/Alfan/Desktop/Data.txt Browse...

Data Preview:

X (integer)	Y (integer)
1	3
4	5
6	5
5	7

Previewing first 50 entries.

Import Options:

Name: Data
Skip: 0

☒ First Row as Name Delimiter: **Whitespace** Escape: None
☒ Trim Spaces Quotes: Default Comment: Default
☒ Open Data Viewer Locale: Configure... NA: Default

Code Preview:

```
library(readr)
Data <- read_delim("C:/Users/Alfan/Desktop/Data.txt",
  " ", escape_double = FALSE, trim_ws = TRUE)
view(Data)
```

Import Cancel

Change delimiter with
whitespace, and click
import

Introduction to R

Read data and write data



The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a search bar. The main workspace is divided into four panes:

- Data Viewer:** Displays a table with 4 rows and 2 columns (X and Y). The data is as follows:

	X	Y
1	1	3
2	4	5
3	6	5
4	5	7
- Environment:** Shows the Global Environment with a single object named 'Data' containing 4 observations of 2 variables.
- Files:** A file explorer showing the current directory structure, including folders like Custom Office Templates, IBM, My Data Sources, My ISO Files, My Tableau Repository, Python Scripts, R, and SafeNet Sentinel.
- Console:** Contains the R code used to read the data:

```
> library(readr)
> Data <- read_delim("C:/Users/Alfan/Desktop/Data.txt",
+ ", ", escape_double = FALSE, trim_ws = TRUE)
Parsed with column specification:
cols(
  x = col_integer(),
  y = col_integer()
)
> view(Data)
>
```

Introduction to R



Read data and write data

Introduction to R



Read data and write data

Introduction to R



Read data and write data

#Function write.table

```
write.table(x, file = "", , quote = TRUE, sep = " ", na = "NA", dec = ".",  
row.names = TRUE, col.names = TRUE)
```

<code>x</code>	the object to be written, preferably a matrix or data frame. If not, it is attempted to coerce <code>x</code> to a data frame.
<code>file</code>	either a character string naming a file or a connection open for writing. "" indicates output to the console.
<code>quote</code>	a logical value (TRUE or FALSE) or a numeric vector. If TRUE, any character or factor columns will be surrounded by double quotes. If a numeric vector, its elements are taken as the indices of columns to quote. In both cases, row and column names are quoted if they are written. If FALSE, nothing is quoted.
<code>sep</code>	the field separator string. Values within each row of <code>x</code> are separated by this string.
<code>na</code>	the string to use for missing values in the data.
<code>dec</code>	the string to use for decimal points in numeric or complex columns: must be a single character.
<code>row.names</code>	either a logical value indicating whether the row names of <code>x</code> are to be written along with <code>x</code> , or a character vector of row names to be written.
<code>col.names</code>	either a logical value indicating whether the column names of <code>x</code> are to be written along with <code>x</code> , or a character vector of column names to be written. See the section on 'CSV files' for the meaning of <code>col.names = NA</code> .

Introduction to R



Read data and write data

#Function write.csv

```
write.csv(x, file = "", , quote = TRUE, sep = " ", na = "NA", dec = ".",  
row.names = TRUE, col.names = TRUE)
```

<code>x</code>	the object to be written, preferably a matrix or data frame. If not, it is attempted to coerce <code>x</code> to a data frame.
<code>file</code>	either a character string naming a file or a connection open for writing. "" indicates output to the console.
<code>quote</code>	a logical value (TRUE or FALSE) or a numeric vector. If TRUE, any character or factor columns will be surrounded by double quotes. If a numeric vector, its elements are taken as the indices of columns to quote. In both cases, row and column names are quoted if they are written. If FALSE, nothing is quoted.
<code>sep</code>	the field separator string. Values within each row of <code>x</code> are separated by this string.
<code>na</code>	the string to use for missing values in the data.
<code>dec</code>	the string to use for decimal points in numeric or complex columns: must be a single character.
<code>row.names</code>	either a logical value indicating whether the row names of <code>x</code> are to be written along with <code>x</code> , or a character vector of row names to be written.
<code>col.names</code>	either a logical value indicating whether the column names of <code>x</code> are to be written along with <code>x</code> , or a character vector of column names to be written. See the section on 'CSV files' for the meaning of <code>col.names = NA</code> .

Introduction to R



Read data and write data

#Example

```
write.table(Data, "D:/Folder/Data.txt", sep=" ", col.names=TRUE, row.names=TRUE,  
quote=FALSE, na="NA")
```

Name of file

```
write.csv(Data, "D:/Folder/Data.csv", sep=" ", col.names=TRUE, row.names=TRUE,  
quote=FALSE, na="NA")
```

Location file will be saved

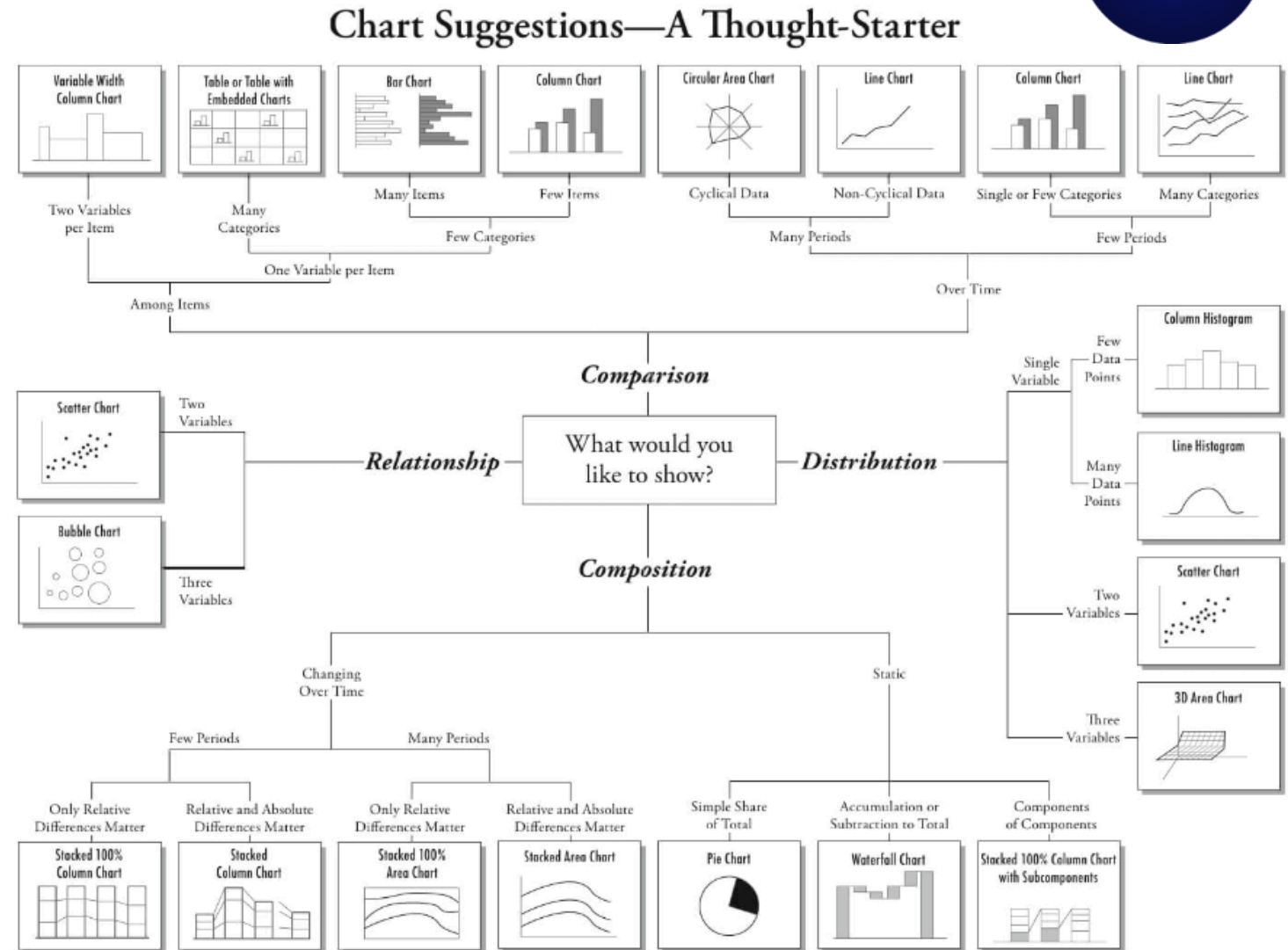
Visualization and Summary Data

Visualization and Summary Data



There are four basic presentation types:

- Comparison
- Composition
- Distribution
- Relationship



Visualization and Summary Data



Titanic data set

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Pa
1	0	3	Braund, Mr. Owen Harris	male	22.00000	1	0
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38.00000	1	0
3	1	3	Heikkinen, Miss. Laina	female	26.00000	0	0
4	1	1	Futelle, Mrs. Jacques Heath (Lily May Peel)	female	35.00000	1	0
5	0	3	Allen, Mr. William Henry	male	35.00000	0	0
6	0	3	Moran, Mr. James	male	29.69912	0	0
7	0	1	McCarthy, Mr. Timothy J	male	54.00000	0	0
8	0	3	Palsson, Master. Gosta Leonard	male	2.00000	3	1
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.00000	0	2
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.00000	1	0
11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.00000	1	1
12	1	1	Bonnell, Miss. Elizabeth	female	58.00000	0	0
13	0	3	Saunderscock, Mr. William Henry	male	20.00000	0	0

survival - Survival (0 = No; 1 = Yes)

class - Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)

name - Name

sex - Sex

age - Age

sibsp - Number of Siblings/Spouses Aboard

parch - Number of Parents/Children Aboard

ticket - Ticket Number

fare - Passenger Fare

cabin - Cabin

embarked - Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

boat - Lifeboat (if survived)

body - Body number (if did not survive and body was recovered)

```
'data.frame': 891 obs. of 12 variables:
 $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int 0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int 3 1 3 1 3 3 1 3 3 2 ...
 $ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",...: 109 191 358 277 16 559 520 629 417
 $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
 $ Age        : num 22 38 26 35 35 ...
 $ SibSp      : int 1 1 0 1 0 0 0 3 0 1 ...
 $ Parch      : int 0 0 0 0 0 0 0 1 2 0 ...
 $ Ticket     : Factor w/ 681 levels "110152","110413",...: 524 597 670 50 473 276 86 396 345 133
 $ Fare       : Factor w/ 247 levels "0","1,108,833",...: 174 183 172 142 210 225 139 71 14 106 ..
 $ Cabin      : Factor w/ 148 levels "", "A10", "A14",...: 1 83 1 57 1 1 131 1 1 1 ...
 $ Embarked   : Factor w/ 4 levels "", "C", "Q", "S": 4 2 4 4 4 3 4 4 4 2 ...
```

Visualization and Summary Data



Histogram

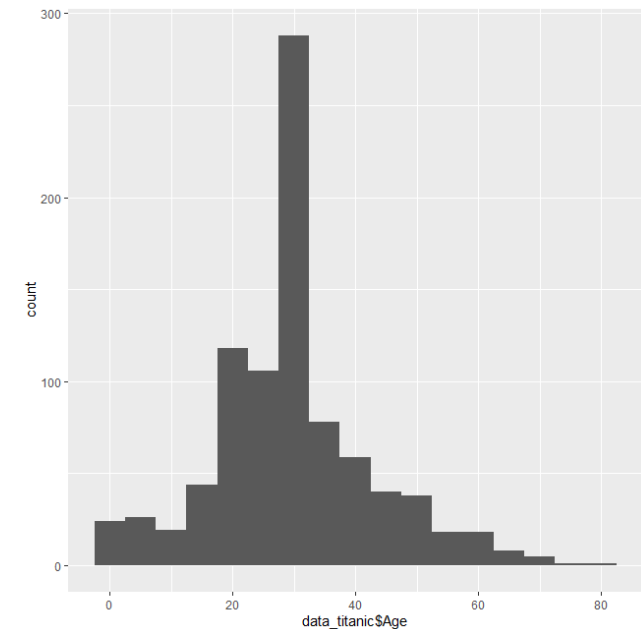
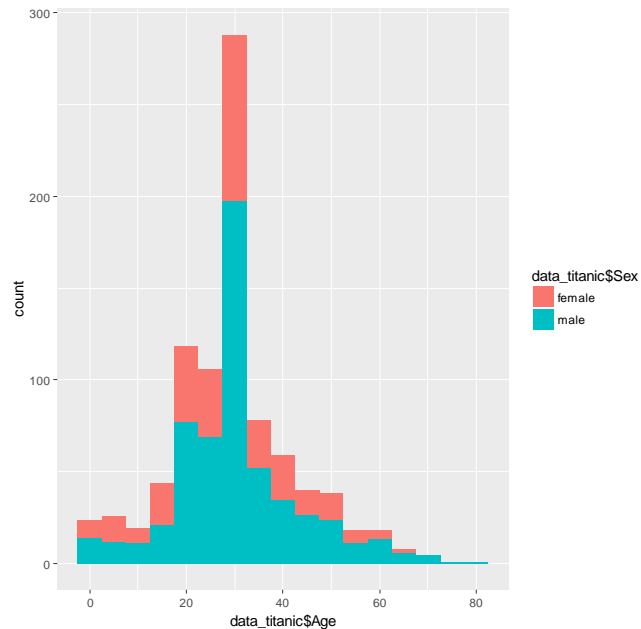
#using Titanic Dataset in R

```
library(ggpubr)
```

```
ggplot(data = data_titanic, mapping = aes(x = data_titanic$Age, fill = data_titanic$Sex),
```

```
palette = c("#00AFBB", "#E7B800")) +  
geom_histogram(binwidth = 5)
```

```
ggplot(data = data_titanic, mapping = aes(x = data_titanic$Age), palette = c("#00AFBB",  
"#E7B800")) + geom_histogram(binwidth = 5)
```

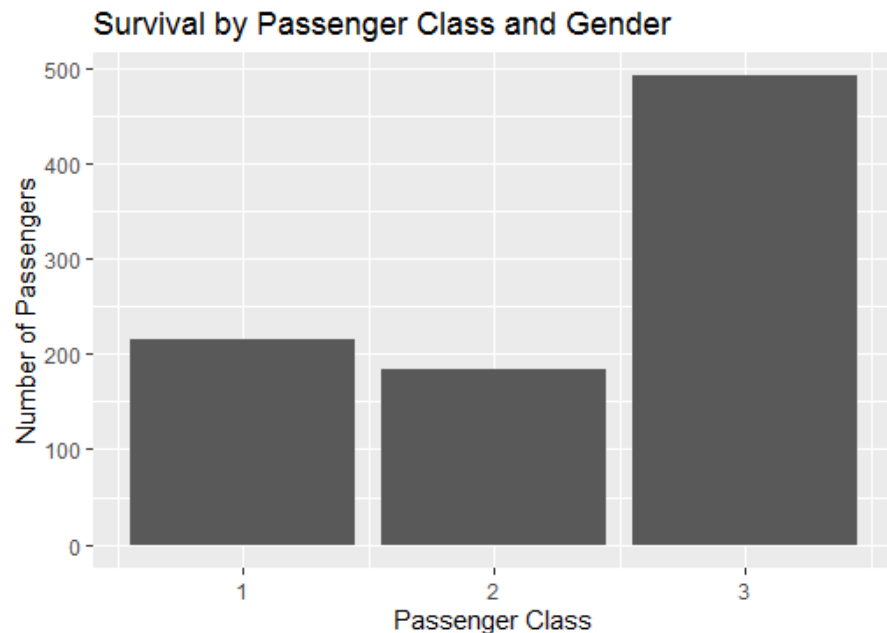
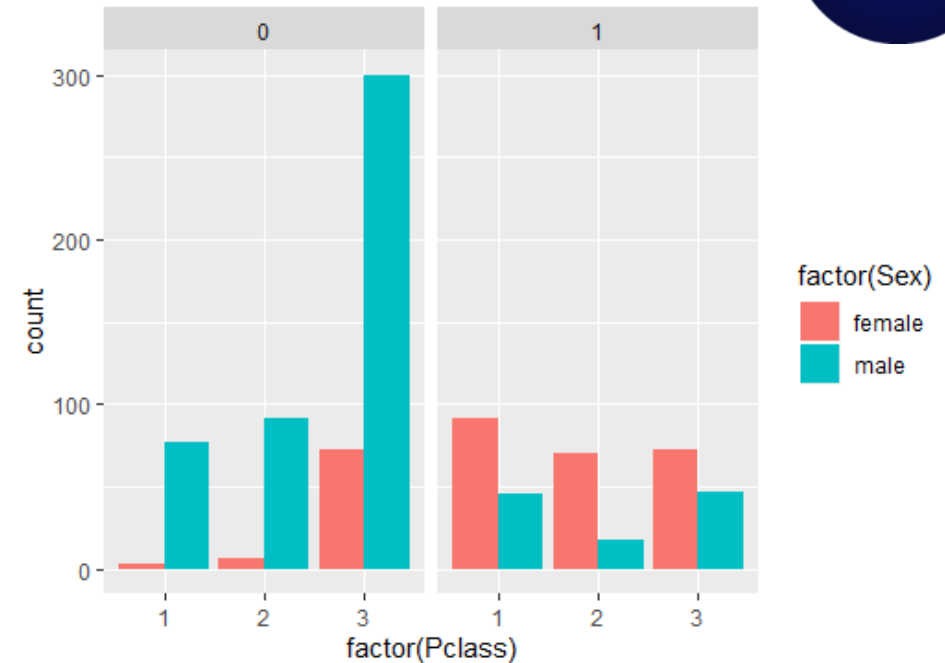


Visualization and Summary Data



Bar Chart

```
ggplot(data_titanic, aes(x=factor(Pclass), fill=
factor(Sex)))+ geom_bar(position="dodge")+
  facet_grid(". ~ Survived")
```

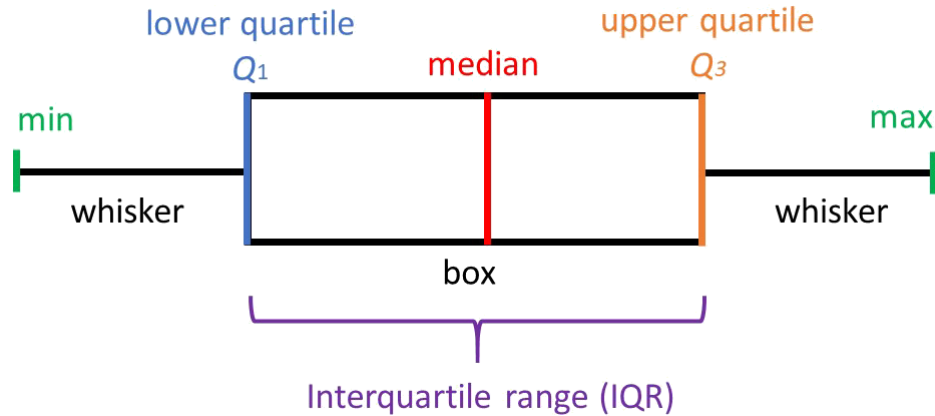


```
barchart <- ggplot(data_titanic, aes(Pclass,
fill=Survived))+geom_bar()
barchart+xlab("Passenger Class")+ylab("Number of
Passengers")+
  ggtitle("Survival by Passenger Class and Gender")+
  scale_fill_discrete(name = "", labels = c("Died",
"Survived"))
```

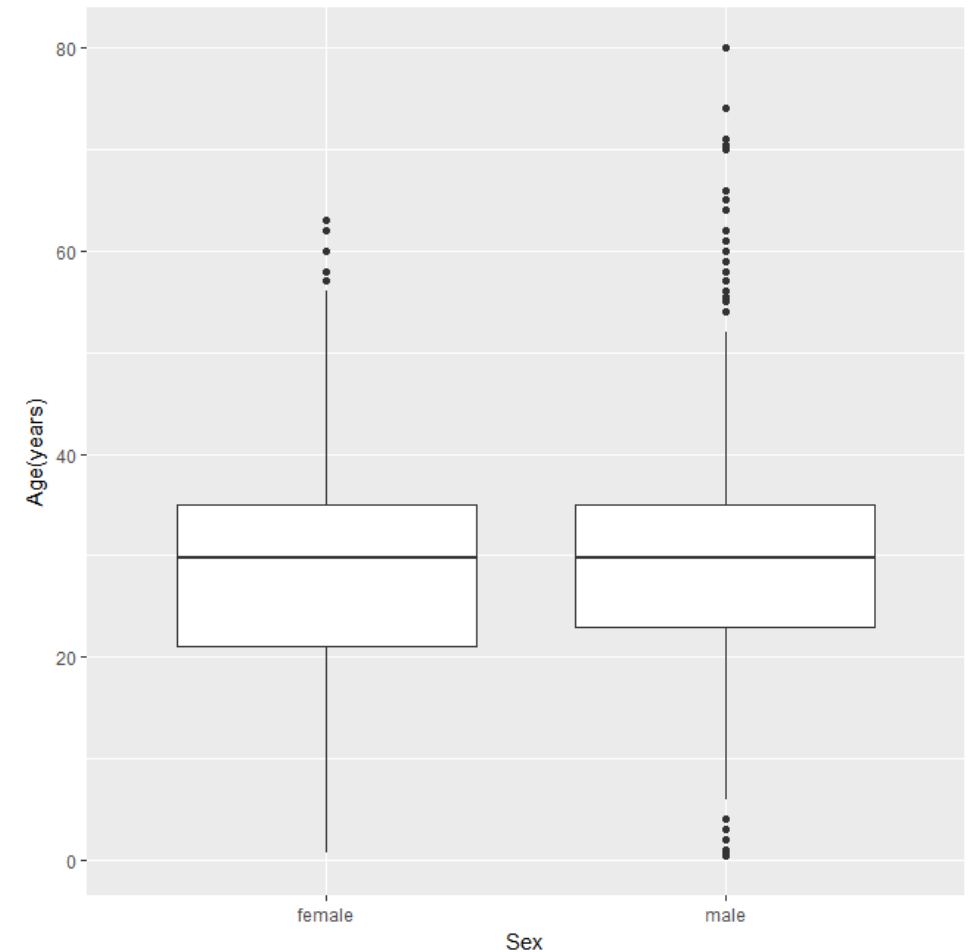
Visualization and Summary Data



Boxplot



```
ggplot(data = data_titanic, mapping = aes(x =  
data_titanic$Sex, y = data_titanic$Age)) +  
  geom_boxplot() +  
  xlab("Sex") + ylab("Age(years)")
```



Visualization and Summary Data



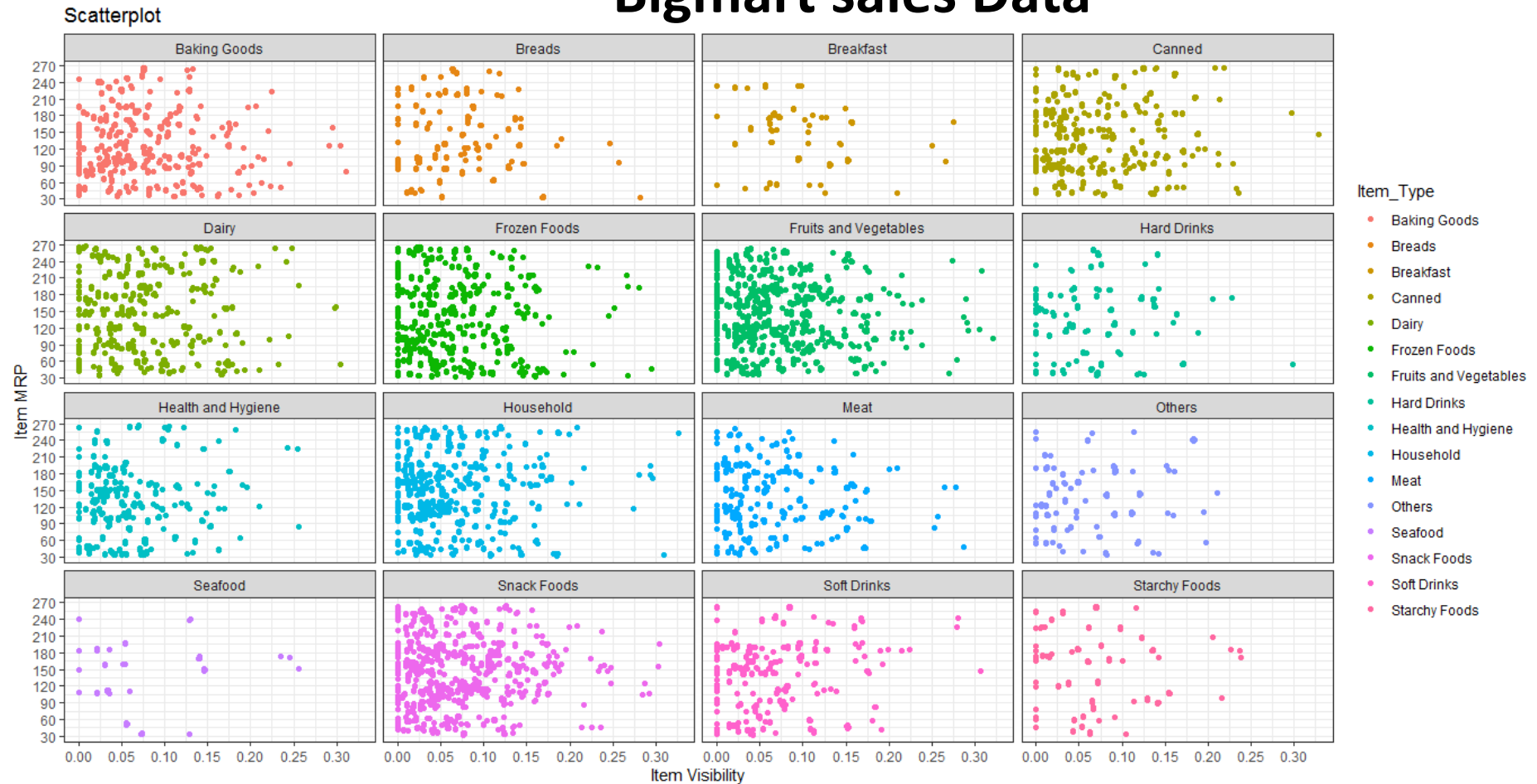
Scatter Plot

```
ggplot(train,
aes(Item_Visibility,
Item_MRP)) +
geom_point(aes(color =
Item_Type)) +
```

```
scale_x_continuous("Item Visibility", breaks
= seq(0,0.35,0.05))+
```

```
scale_y_continuous("Item MRP", breaks =
seq(0,270,by = 30))+
theme_bw() +
labs(title="Scatterplot") + facet_wrap(~
Item_Type)
```

Bigmart sales Data



Visualization Data and Summary Data



Let's make a simple story

Visualization Data and Summary Data



Iris Dataset

About the Iris Data Set

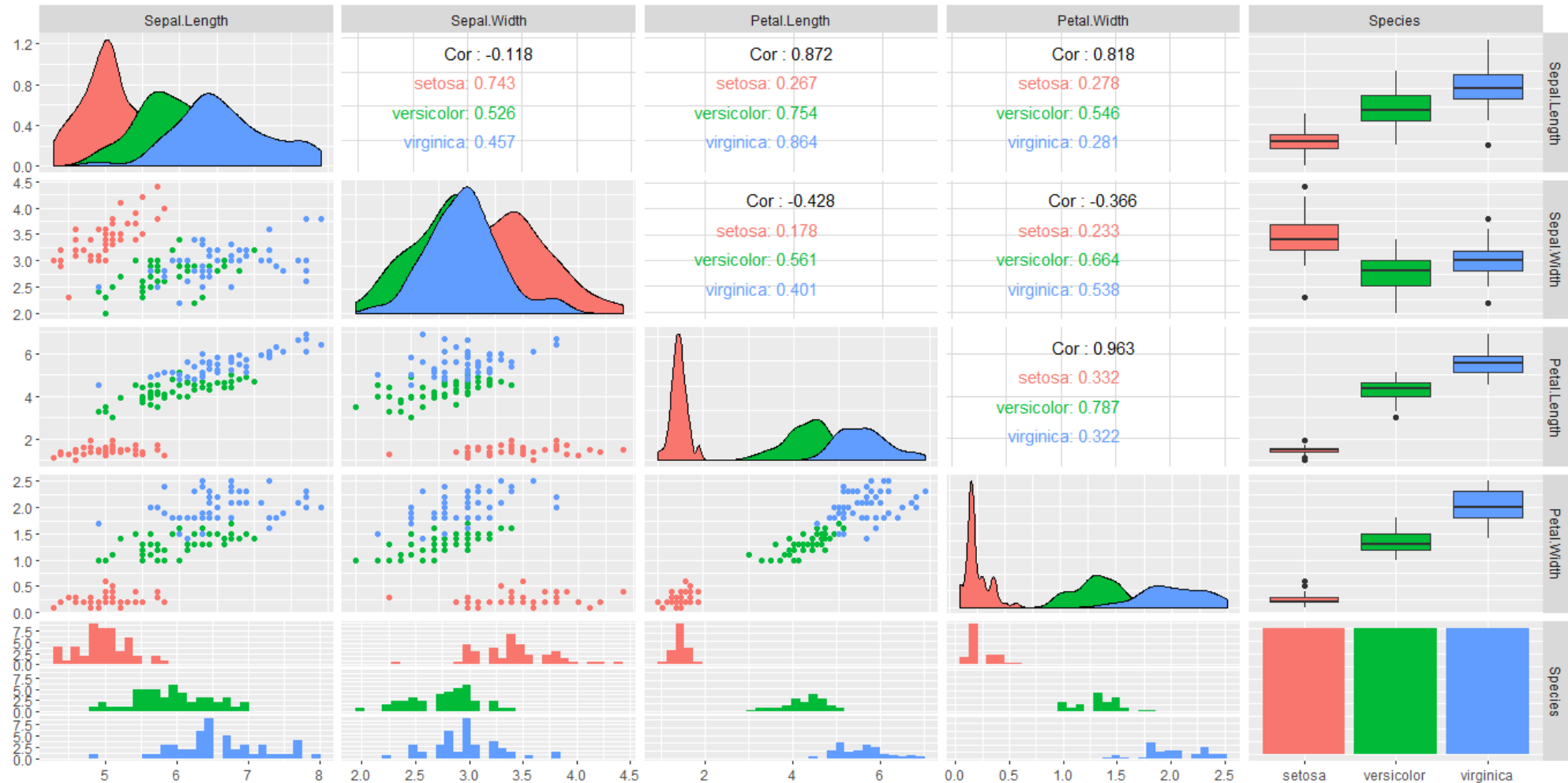
- * The Iris Data Set is a multivariate data set used by R. A Fisher in his 1936 paper "The use of multiple measurements in taxonomic problems" as an example of linear discriminate analysis.
- * The data was collected by Edgar Andersen in 1935 to quantify the morphologic variation of Iris flowers of three related species.
- * Two of the species were collected in the Gaspé Peninsula, Canada, in one pasture, picked the same day, measured at the same time by the same person using the same equipment
- * The Iris data set contains 150 random observations and 5 variables (one categorical and 4 numeric) from three iris species, setosa, versicolor, and virginica.
- * There are 50 observations from each of the three iris species, measuring sepal length, sepal width, petal length and petal width, all numeric values in centimeters.
- * There is no missing data.

Visualization Data and Summary Data

`view(iris)`

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

Visualization Data and Summary Data



```
ggpairs(iris[,c(1:5)],ggplot2::aes(colour=Species))
```

Visualization Data and Summary Data

```
Setosa=subset(iris,Species=="setosa")  
summary(Setosa[, -5])  
Versicolor=subset(iris,Species=="versicolor")  
summary(Versicolor[, -5])  
Virginica=subset(iris,Species=="virginica")  
summary(Virginica[, -5])
```

GitHub



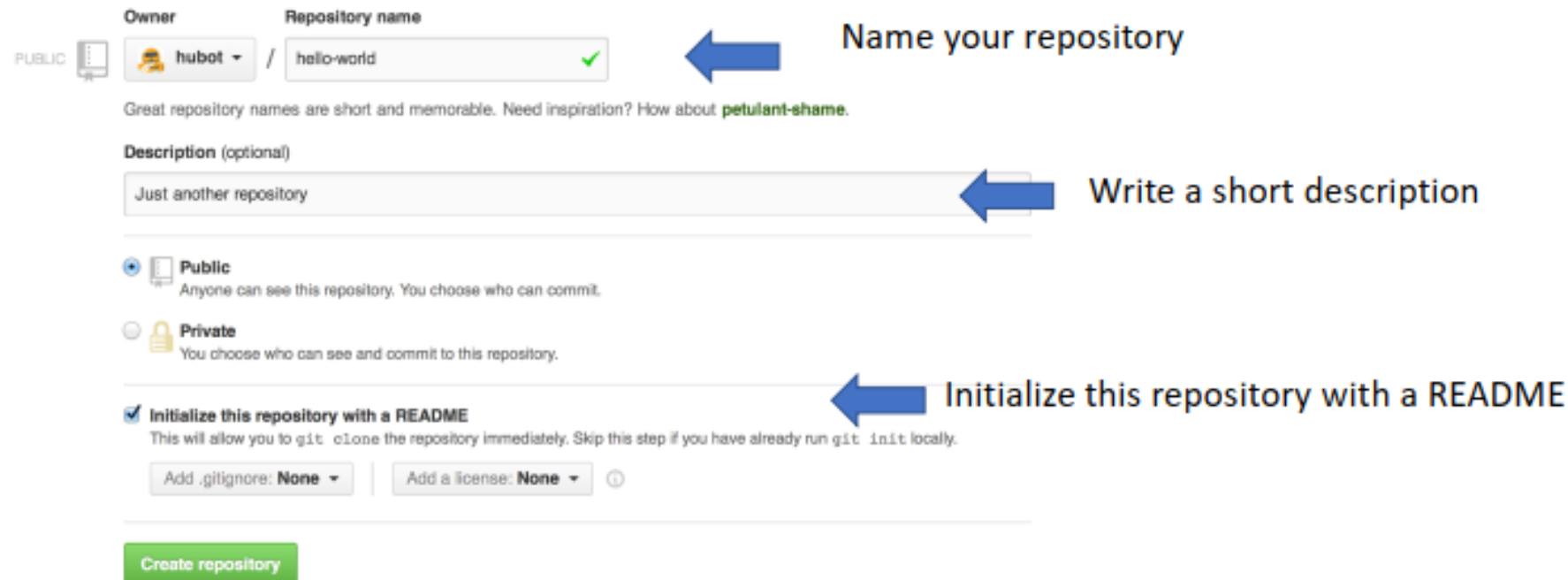
What is GitHub?

code hosting platform
for version control and
collaboration. It lets you
and others work
together on projects
from anywhere.

A screenshot of the GitHub sign-up page. The page has a dark background with a light-colored sign-up form on the right. The form contains fields for Username, Email, and Password, each with a placeholder text. Below the Password field is a green button labeled "Sign up for GitHub". At the bottom of the form, there is a line of text stating: "By clicking 'Sign up for GitHub', you agree to our [terms of service](#) and [privacy statement](#). We'll occasionally send you account related emails." The main content area on the left of the form features the GitHub logo, navigation links (Features, Business, Explore, Marketplace, Pricing), a search bar, and a "Sign in or Sign up" link. The main heading "Built for developers" is prominently displayed, followed by a paragraph describing GitHub as a development platform inspired by the way you work, from open source to business, where you can host and review code, manage projects, and build software alongside 30 million developers.

Repository

A **repository** is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs.



The screenshot shows the GitHub repository creation interface. It includes a 'PUBLIC' toggle, an 'Owner' dropdown set to 'hubot', and a 'Repository name' field containing 'hello-world' with a green checkmark. A blue arrow points from the text 'Name your repository' to the repository name field. Below this is a 'Description (optional)' text area containing 'Just another repository', with a blue arrow pointing from the text 'Write a short description' to it. The 'Visibility' section shows 'Public' selected with the subtext 'Anyone can see this repository. You choose who can commit.' and 'Private' unselected with the subtext 'You choose who can see and commit to this repository.' The 'Initialize this repository with a README' checkbox is checked, with a blue arrow pointing from the text 'Initialize this repository with a README' to it. Below this are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None'. At the bottom is a green 'Create repository' button.

Owner: hubot / Repository name: hello-world ✓

Great repository names are short and memorable. Need inspiration? How about [petulant-shame](#).

Description (optional): Just another repository

Public: Anyone can see this repository. You choose who can commit.

Private: You choose who can see and commit to this repository.

Initialize this repository with a README: This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: None | Add a license: None ⓘ

Create repository

Branch

Branching is the way to work on different versions of a repository at one time.



Have you ever saved different versions of a file? Something like:



- Laporan.docx
- Laporan-revisi-1.docx
- Laporan-final.docx

Branches accomplish similar goals in GitHub repositories.


Branch


Just another repository — Edit


 1 commit
  1 branch


branch: **master** ▾
hello-world / 

Initial commit

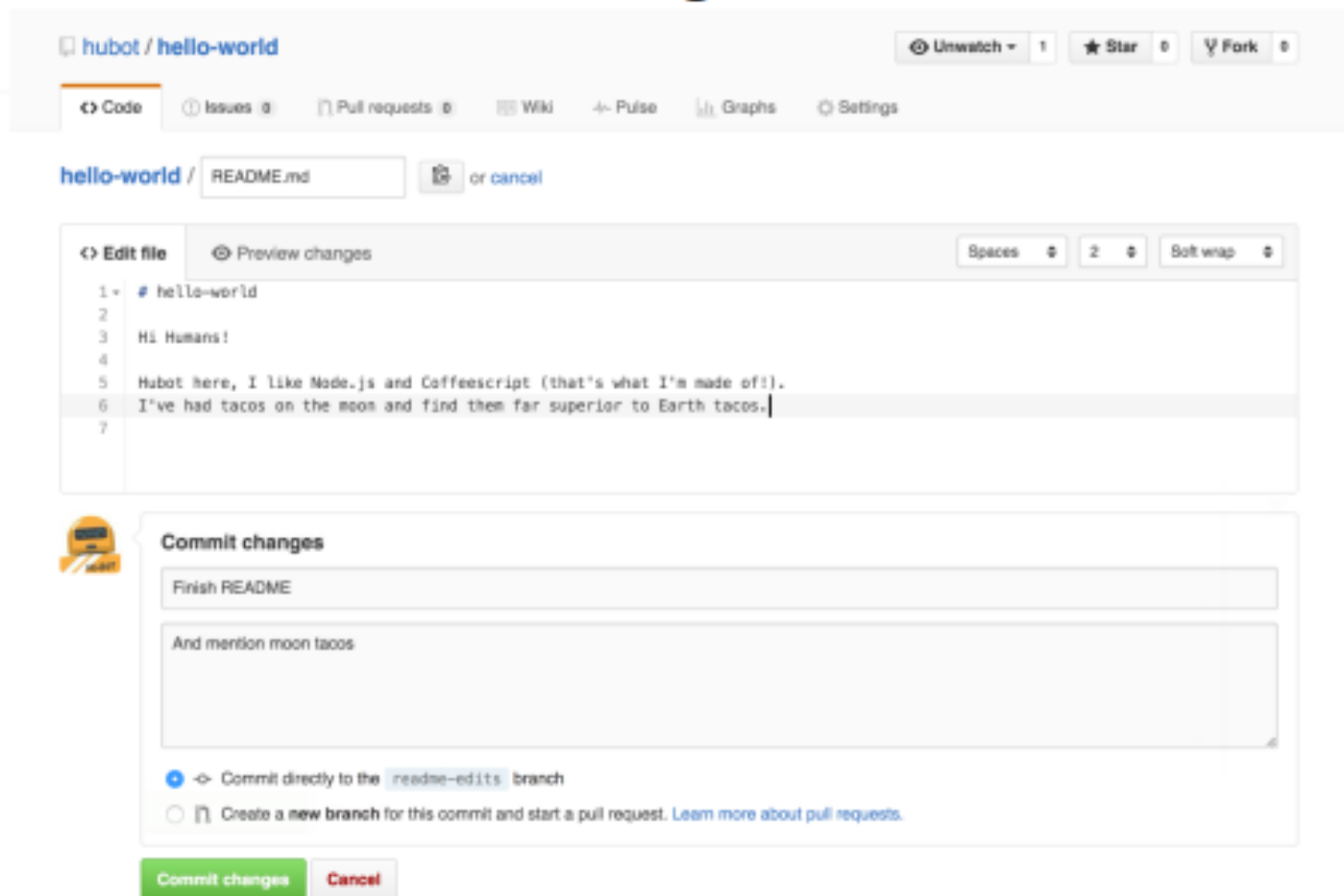
 **hubot** authored just now

 [README.md](#)
Initial

 **README.md**

Make and commit changes

On GitHub, saved changes are called *commits*.



The screenshot shows the GitHub interface for a repository named 'hubot / hello-world'. The 'Code' tab is selected, and the 'README.md' file is being edited. The file content is as follows:

```
1 # hello-world
2
3 Hi Humans!
4
5 Hubot here, I like Node.js and Coffeescript (that's what I'm made of!).
6 I've had tacos on the moon and find them far superior to Earth tacos.
7
```

Below the editor, the 'Commit changes' section is visible. It contains a text input field with the message 'Finish README' and a larger text area with the message 'And mention moon tacos'. The commit options are:

- ☒ Commit directly to the `readme-edits` branch
- ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

At the bottom, there are two buttons: 'Commit changes' (green) and 'Cancel' (red).

These changes will be made to just the README file on your branch, so now this branch contains content that's different from master.