Introduction to Data Mining and Analytics

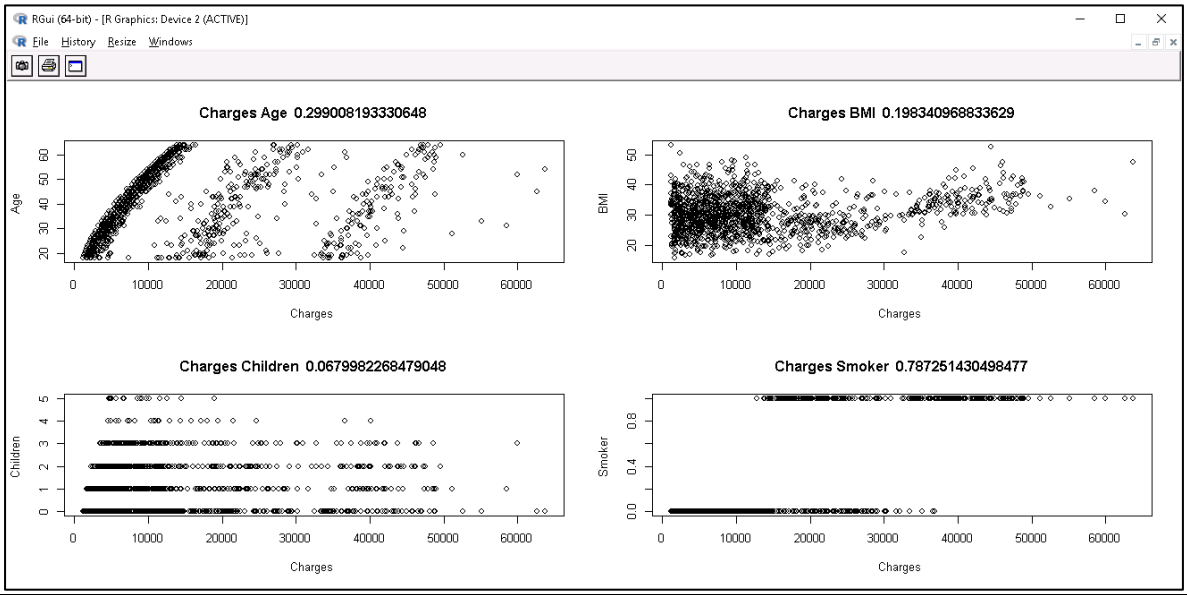
Errata Sheet

Page: 10
Location: first code block

Current Text	***Corrected Text***
corr <- cor(df$charges, df$age) title <- paste("Charges Age ", sprintf("%s", corr)) plot(df$charges, df$bmi, main=title, xlab="Charges ", ylab="Age ") corr <- cor(df$charges, df$bmi) title <- paste("Charges BMI ", sprintf("%s", corr)) plot(df$charges, df$bmi, main=title, xlab="Charges ", ylab="BMI ")	corr <- cor(df$charges, df$age) title <- paste("Charges Age ", sprintf("%s", corr)) plot(df$charges, df$age, main=title, xlab="Charges ", ylab="Age ") corr <- cor(df$charges, df$bmi) title <- paste("Charges BMI ", sprintf("%s", corr)) plot(df$charges, df$bmi, main=title, xlab="Charges ", ylab="BMI ")

Page: 11
Location: Figure 1.7

Corrected Screenshot



Page: 152
Location: first paragraph

Current Text	*Corrected Text*
Within Tableau, a single visualization is called a sheet (some analysts review to a sheet as a view).	Within Tableau, a single visualization is called a sheet (some analysts refer to a sheet as a view).

Page: 188
Location: Figure 5.36 and 5.37

Corrected Equations
$$\text{DEVSQ} = \sum_{i=1}^{n}(x_i - \overline{x})^2$$
$$\text{AVEDEV} = \frac{\sum_{i=1}^{n}\lvert x_i - \overline{x}\rvert}{n}$$

Page: 230
Location: first paragraph

Current Text	*Corrected Text*
For example, the following SELECT query will sort sales data by region and then sort that data by employee last name:	For example, the following SELECT query will sort employee data by region and then by last name:

Page: 257
Location: first code block

Current Text	***Corrected Text***
SELECT * FROM Customers JOIN Orders WHERE Customers.CustomerID = Orders.CustomerID	SELECT * FROM Customers JOIN Orders ON Customers.CustomerID = Orders.CustomerID

Page: 257
Location: second code block

Current Text	***Corrected Text***
SELECT * FROM Customers c JOIN Orders o WHERE c.CustomerID = o.CustomerID	SELECT * FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID

Page: 258
Location: first code block

Current Text	***Corrected Text***
SELECT c.Lastname, o.OrderID FROM Customers c INNER JOIN Orders o WHERE c.CustomerID = o.CustomerID	SELECT c.Lastname, o.OrderID FROM Customers c INNER JOIN Orders o ON c.CustomerID = o.CustomerID

Page: 258
Location: second code block

Current Text	*Corrected Text*
SELECT c.Lastname, o.OrderID FROM Customers c LEFT JOIN Orders o WHERE c.CustomerID = o.CustomerID	SELECT c.Lastname, o.OrderID FROM Customers c LEFT JOIN Orders o ON c.CustomerID = o.CustomerID

Page: 259
Location: first code block

Current Text	*Corrected Text*
LEFT JOIN Orders o WHERE c.CustomerID = o.CustomerID	LEFT JOIN Orders o ON c.CustomerID = o.CustomerID

Page: 260
Location: first code block

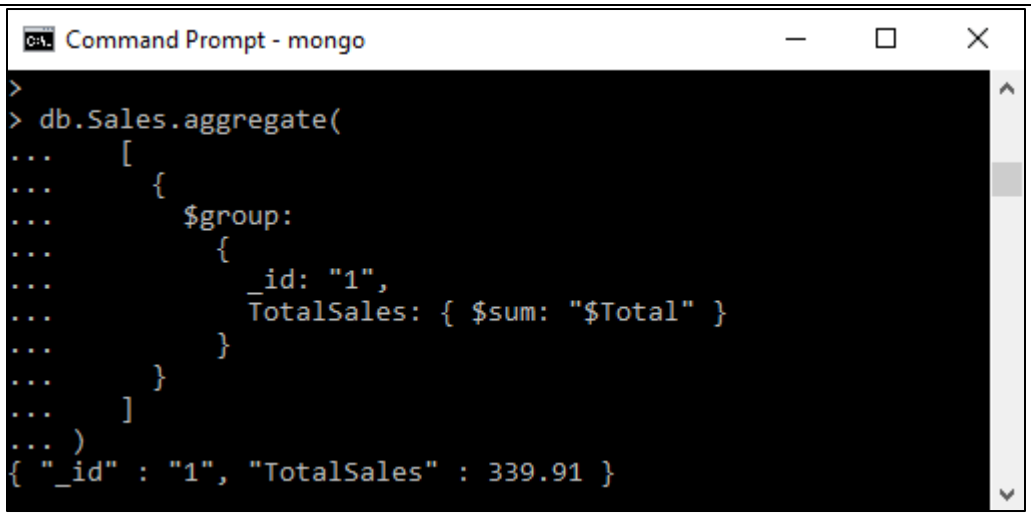
Current Text	*Corrected Text*
RIGHT JOIN Orders o WHERE c.CustomerID = o.CustomerID WHERE CustomerID = NULL	RIGHT JOIN Orders o ON c.CustomerID = o.CustomerID WHERE CustomerID IS NULL

Page: 313
Location: only code block

Current Text	*Corrected Text*

`db.Products.aggregate(`
`  [`
`  {`
`  $group:`
`  {`
`  _id: "1",`
`  TotalSales: {$sum: "$Price"}`
`  }`
`  }`
`  ]`
`)`
	`db.Sales.aggregate(`
	`  [`
	`  {`
	`  $group:`
	`  {`
	`  _id: "1",`
	`  TotalSales: {$sum: "$Total"}`
	`  }`
	`  }`
	`  ]`
	`)`

Page: 314
Location: figure 7.30

Corrected Screenshot


Page: 363
Location: First code block under *Leveraging Python's Built-In Functions*

Current Text	***Corrected Text***
print('Power of 5 raised to 2 is', pow(5, 2))	print('5 raised to the power of 2 is', pow(5, 2))

Page: 364
Location: first code block

Current Text	***Corrected Text***
Power of 5 raised to 2 is 25	5 raised to the power of 2 is 25

Page: 366
Location: first code block

Current Text	***Corrected Text***
16 1 : 16 19	16 1 16 19

Page: 397
Location: fifth code block

Current Text	***Corrected Text***
import packageName	library(packageName)

Page: 415
Location: second code block

Current Text	*Corrected Text*
SELECT COUNT(*) FROM SensorTable WHERE SensorValue IS NULL	SELECT COUNT(*) FROM Sensor WHERE SensorValue IS NULL

Page: 417
Location: second code block

Current Text	*Corrected Text*
SELECT COUNT(*) AS 'Distinct Record Count' FROM Customers	SELECT DISTINCT COUNT(*) AS 'Distinct Record Count' FROM (Select DISTINCT * FROM Customers) c

---

Page: 418
Location: second paragraph

Current Text	*Corrected Text*
The following query performs a similar operation on the Sensors table to display duplicate records:	The following query performs a similar operation on the Sensor table to display duplicate records:

Page: 419
Location: figure 9.8 caption

Current Text	***Corrected Text***
Displaying duplicate records within the Sensors table	Displaying duplicate records within the Sensor table

---

Page: 473
Location: first paragraph

Current Text	***Corrected Text***
The following Python script, ShowNoise.py, displays the noise values identified by DBSCAN:	The following Python script, ShowOutliers.py, displays the noise values identified by DBSCAN:

Page: 499
Location: second code block

Current Text	***Corrected Text***
## A 3-nearest neighbors model with no normalization	## A 5-nearest neighbors model with no normalization

---

Page: 507
Location: second paragraph

Current Text	***Corrected Text***
The following Python script, LogitisticRegressionIris.py, uses the model to predict iris flower types:	The following Python script, LogisticRegressionIris.py, uses the model to predict iris flower types:

Page: 522
Location: second paragraph, second sentence

Current Text	***Corrected Text***
Then open a text editor, such as Notepad, and press Ctrl+V to paste the data. Save the data file to your disk using the name Zoo.csv.	Then open a text editor, such as Notepad, and press Ctrl+V to paste the data. Label the data by inserting the following text, on its own line, at the top of the file: *animal_name,hair,feathers,eggs,milk,airborne,aquatic,predator,toothed,backbone,breathes,venomous,fins,legs,tail,domestic,catsize,class_type* Save the data file to your disk using the name Zoo.csv

---

Page: 542
Location: second code block

Current Text	***Corrected Text***
C:\> python SeattleHousing.pyd	C:\> python SeattleHousing.py

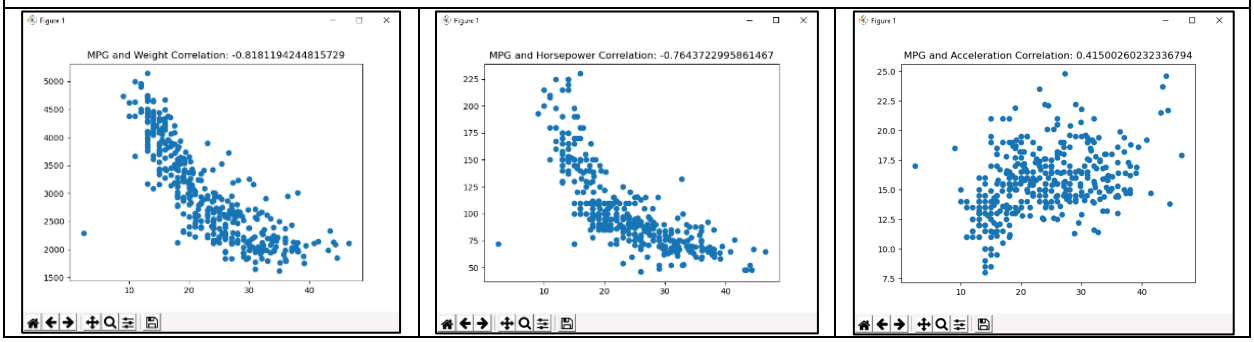
Page: 552
Location: first code block

Current Text	*Corrected Text*
y = np.array([2,3,9,12,15,18,19,20])	y = np.array([2,3,9,13,27,84,105,169])

Page: 576
Location: second code block

Current Text	*Corrected Text*
coefs = np.corrcoef(data['mpg'], data['weight']) plt.scatter(data['mpg'], data['weight']) plt.title('MPG and Weight Correlation: ' + str(coefs[0,1])) plt.show() coefs = np.corrcoef(data['mpg'], data['horsepower']) plt.scatter(data['mpg'], data['horsepower']) plt.title('MPG and Weight Horsepower: ' + str(coefs[0,1])) plt.show() coefs = np.corrcoef(data['mpg'], data['acceleration']) plt.scatter(data['mpg'], data['acceleration']) plt.title('MPG and Weight Acceleration: ' + str(coefs[0,1])) plt.show()	coefs = np.corrcoef(data['mpg'], data['weight']) plt.scatter(data['mpg'], data['weight']) plt.title('MPG and Weight Correlation: ' + str(coefs[0,1])) plt.show() coefs = np.corrcoef(data['mpg'], data['horsepower']) plt.scatter(data['mpg'], data['horsepower']) plt.title('MPG and Horsepower Correlation: ' + str(coefs[0,1])) plt.show() coefs = np.corrcoef(data['mpg'], data['acceleration']) plt.scatter(data['mpg'], data['acceleration']) plt.title('MPG and Acceleration Correlation: ' + str(coefs[0,1])) plt.show()

Page: 577
Location: figure 13.12

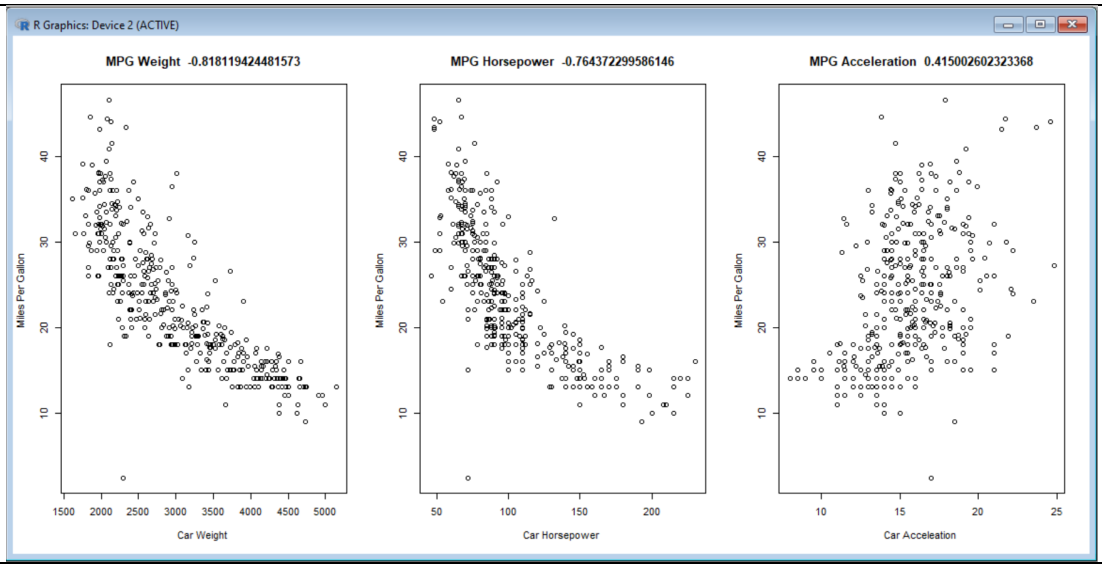
Corrected Screenshot


Page: 578
Location: third code block

Current Text	*Corrected Text*

corr <- cor(dfmpg, dfweight)
title <- paste("MPG Weight ", sprintf("%s", corr))
plot(df$weight, df$mpg, main=title,
  xlab="Car Weight ", ylab="Miles Per Gallon ")
 	
corr <- cor(dfmpg, dfhorsepower)
title <- paste("MPG Horsepower ", sprintf("%s", corr))
plot(df$weight, df$mpg, main=title,
  xlab="Car Horsepower ", ylab="Miles Per Gallon ")
 	
corr <- cor(dfmpg, dfacceleration)
title <- paste("MPG Acceleration ", sprintf("%s", corr))
plot(df$weight, df$mpg, main=title,
  xlab="Car Acceleation ", ylab="Miles Per Gallon ")
	corr <- cor(dfmpg, dfweight)
	title <- paste("MPG Weight ", sprintf("%s", corr))
	plot(df$weight, df$mpg, main=title,
	  xlab="Car Weight ", ylab="Miles Per Gallon ")
	
	corr <- cor(dfmpg, dfhorsepower)
	title <- paste("MPG Horsepower ", sprintf("%s", corr))
	plot(df$horsepower, df$mpg, main=title,
	  xlab="Car Horsepower ", ylab="Miles Per Gallon ")
	
	corr <- cor(dfmpg, dfacceleration)
	title <- paste("MPG Acceleration ", sprintf("%s", corr))
	plot(df$acceleration, df$mpg, main=title,
	  xlab="Car Acceleation ", ylab="Miles Per Gallon ")

Page: 579
Location: figure 13.13

Corrected Screenshot


Page: 601
Location: third code block

Current Text	*Corrected Text*
After you install these two applications, use PIP to install the Facial_Recognition and DLib modules:	After you install these two applications, use PIP to install the face_recognition and DLib modules:

Page: 601
Location: third code block

Current Text	*Corrected Text*
C:\> pip install Facial_Recognition	C:\> pip install face_recognition