

# Mathematics and Programming for Machine Learning with R for Business and Engineering

William B. Cluster

---

*Contents*

---

1 -Functions Tutorial .....	16
1.2 Functions .....	19
Practice 1.2.2.....	20
1.3 Parameter versus argument .....	21
Practice 1.3.3.....	21
1.4 Argument Order and Parameter Names.....	23
Practice 1.4.4.....	23
1.5 Environments .....	23
1.6 Scope.....	<b>Error! Bookmark not defined.</b>
Practice 1.6.5.....	<b>Error! Bookmark not defined.</b>
2 -Logic and R .....	<b>Error! Bookmark not defined.</b>
2.1 Logic .....	<b>Error! Bookmark not defined.</b>
2.2 Statements .....	<b>Error! Bookmark not defined.</b>
Practice 2.2.1.....	<b>Error! Bookmark not defined.</b>
2.3 Boolean Data Type .....	<b>Error! Bookmark not defined.</b>
Practice 2.3.2.....	<b>Error! Bookmark not defined.</b>
Practice 2.3.3.....	<b>Error! Bookmark not defined.</b>

2.4 Compound Statements .....	Error! Bookmark not defined.
2.5 Connectives.....	Error! Bookmark not defined.
Practice 2.5.4.....	Error! Bookmark not defined.
2.6 'or' -Disjunction.....	Error! Bookmark not defined.
Practice 2.6.5.....	Error! Bookmark not defined.
2.7 Negation.....	Error! Bookmark not defined.
Practice 2.7.6.....	Error! Bookmark not defined.
2.8 Logical Equivalence .....	Error! Bookmark not defined.
Practice 2.8.7.....	Error! Bookmark not defined.
2.9 Implementing Logic in the Context of a Dataframe.....	Error! Bookmark not defined.
2.10 NA in Truth Table .....	Error! Bookmark not defined.
2.11 Conclusion.....	Error! Bookmark not defined.
3 Sets with R: Building the tools .....	Error! Bookmark not defined.
3.1 Sets.....	Error! Bookmark not defined.
3.2 Venn Diagrams .....	Error! Bookmark not defined.
3.3 Cardinality of a set .....	Error! Bookmark not defined.
Practice 3.3.1.....	Error! Bookmark not defined.
Practice 3.3.2.....	Error! Bookmark not defined.
3.4 Implementing the subset function in R.....	Error! Bookmark not defined.
Practice 3.4.3.....	Error! Bookmark not defined.
Practice 3.4.4.....	Error! Bookmark not defined.
3.5 Equality of sets.....	Error! Bookmark not defined.
Practice 3.5.5.....	Error! Bookmark not defined.
3.6 Empty Set: {}.....	Error! Bookmark not defined.
3.7 Intersection: $A \cap B$ .....	Error! Bookmark not defined.
Practice 3.7.6.....	Error! Bookmark not defined.
3.8 Union $A \cup B$ .....	Error! Bookmark not defined.
Practice 3.8.7.....	Error! Bookmark not defined.
3.9 Complement $A'$ (Absolute complement).....	Error! Bookmark not defined.
Practice 3.9.8.....	Error! Bookmark not defined.
3.10 Implementation of Complement without using <code>setdiff()</code> .....	Error! Bookmark not defined.

3.11 Algebraic Properties of Sets .....	<b>Error! Bookmark not defined.</b>
3.12 Notes on the reasoning behind these laws.....	<b>Error! Bookmark not defined.</b>
3.13 Testing some of the properties.....	<b>Error! Bookmark not defined.</b>
Practice 3.13.9.....	<b>Error! Bookmark not defined.</b>
4 Probability .....	<b>Error! Bookmark not defined.</b>
4.1 Dealing with Missing Values in a Dataset .....	<b>Error! Bookmark not defined.</b>
4.2 Experiment, Outcome, Sample Space.....	<b>Error! Bookmark not defined.</b>
4.3 Examples of Experiments .....	<b>Error! Bookmark not defined.</b>
4.4 Sample Space .....	<b>Error! Bookmark not defined.</b>
4.5 Observational units versus outcomes.....	<b>Error! Bookmark not defined.</b>
4.6 Events.....	<b>Error! Bookmark not defined.</b>
4.7 Compound Event using Set Operations .....	<b>Error! Bookmark not defined.</b>
4.8 Union of Two Events .....	<b>Error! Bookmark not defined.</b>
4.9 The union of 2 events is an event .....	<b>Error! Bookmark not defined.</b>
4.10 Union of several events .....	<b>Error! Bookmark not defined.</b>
4.11 The Ellipsis.....	<b>Error! Bookmark not defined.</b>
4.12 Intersection .....	<b>Error! Bookmark not defined.</b>
4.13 Complement or Negation of an Event .....	<b>Error! Bookmark not defined.</b>
Practice 4.13.1.....	<b>Error! Bookmark not defined.</b>
4.14 Probability Definition .....	<b>Error! Bookmark not defined.</b>
Practice 4.14.2.....	<b>Error! Bookmark not defined.</b>
4.15 Properties for probability.....	<b>Error! Bookmark not defined.</b>
4.16 Probability of mutually exclusive events. ....	<b>Error! Bookmark not defined.</b>
Practice 4.16.3.....	<b>Error! Bookmark not defined.</b>
Practice 4.16.4.....	<b>Error! Bookmark not defined.</b>
4.17 Conditional Probability .....	<b>Error! Bookmark not defined.</b>
4.18 Additional terminology .....	<b>Error! Bookmark not defined.</b>
4.19 Independent Events .....	<b>Error! Bookmark not defined.</b>
4.20 Examples .....	<b>Error! Bookmark not defined.</b>
4.21 Conditional Independence.....	<b>Error! Bookmark not defined.</b>
Practice 4.21.5.....	<b>Error! Bookmark not defined.</b>
Practice 4.21.6.....	<b>Error! Bookmark not defined.</b>

4.22 Bayes Theorem .....	<b>Error! Bookmark not defined.</b>
4.23 Statement of Bayes Theorem .....	<b>Error! Bookmark not defined.</b>
4.24 Partition of a set.....	<b>Error! Bookmark not defined.</b>
Practice 4.24.7.....	<b>Error! Bookmark not defined.</b>
4.25 Proof of Bayes Theorem .....	<b>Error! Bookmark not defined.</b>
4.26 Anoter version of Bayes Theorem .....	<b>Error! Bookmark not defined.</b>
Practice 4.26.8.....	<b>Error! Bookmark not defined.</b>
5 Naïve Rule .....	<b>Error! Bookmark not defined.</b>
5.1 A Sample Data Set.....	<b>Error! Bookmark not defined.</b>
5.2 Labeled Data .....	<b>Error! Bookmark not defined.</b>
Practice 5.2.1.....	<b>Error! Bookmark not defined.</b>
5.3 Notation for a model.....	<b>Error! Bookmark not defined.</b>
5.4 Estimation, Prediction, Classification.....	<b>Error! Bookmark not defined.</b>
Practice 5.4.2.....	<b>Error! Bookmark not defined.</b>
5.5 Naïve Rule -the most rudimentary model .....	<b>Error! Bookmark not defined.</b>
Practice 5.5.3.....	<b>Error! Bookmark not defined.</b>
5.6 R implementation of the Naïve Rule.....	<b>Error! Bookmark not defined.</b>
Practice 5.6.4.....	<b>Error! Bookmark not defined.</b>
Practice 5.6.5.....	<b>Error! Bookmark not defined.</b>
Practice 5.6.6.....	<b>Error! Bookmark not defined.</b>
5.7 Levels.....	<b>Error! Bookmark not defined.</b>
5.8 Naïve Rule Implementation and Explanation .....	<b>Error! Bookmark not defined.</b>
Practice 5.8.7.....	<b>Error! Bookmark not defined.</b>
6 Complete Bayes .....	<b>Error! Bookmark not defined.</b>
6.1 Classification Task in Data Science.....	<b>Error! Bookmark not defined.</b>
6.2 Complete Bayes Classifier .....	<b>Error! Bookmark not defined.</b>
6.3 Weakness in the Complete Bayes.....	<b>Error! Bookmark not defined.</b>
Practice 6.3.1.....	<b>Error! Bookmark not defined.</b>
6.4 Complete Bayes Implementation in R .....	<b>Error! Bookmark not defined.</b>
Practice 6.4.2.....	<b>Error! Bookmark not defined.</b>
Practice 6.4.3.....	<b>Error! Bookmark not defined.</b>
Practice 6.4.4.....	<b>Error! Bookmark not defined.</b>

Tips:.....	<b>Error! Bookmark not defined.</b>
Practice 6.4.5.....	<b>Error! Bookmark not defined.</b>
6.5 Which level is most likely? .....	<b>Error! Bookmark not defined.</b>
6.6 <i>Side note</i> on the use of \$ sign .....	<b>Error! Bookmark not defined.</b>
Practice 6.6.6.....	<b>Error! Bookmark not defined.</b>
6.7 Testing the function with various instances. ....	<b>Error! Bookmark not defined.</b>
Practice 6.7.7.....	<b>Error! Bookmark not defined.</b>
6.8 Complete Bayes Implementation 2 - <i>a more robust version</i> .....	<b>Error! Bookmark not defined.</b>
Practice 6.8.8.....	<b>Error! Bookmark not defined.</b>
6.9 Manual preprocessing of the data set .....	<b>Error! Bookmark not defined.</b>
6.10 Debugging in R .....	<b>Error! Bookmark not defined.</b>
Practice 6.10.9.....	<b>Error! Bookmark not defined.</b>
6.11 A Short Tutorial on <code>paste0()</code> and <code>eval(parse())</code> .....	<b>Error! Bookmark not defined.</b>
Practice 6.11.10.....	<b>Error! Bookmark not defined.</b>
6.12 <code>eval(parse())</code> .....	<b>Error! Bookmark not defined.</b>
Practice 6.12.11.....	<b>Error! Bookmark not defined.</b>
Practice 6.12.12.....	<b>Error! Bookmark not defined.</b>
6.13 Returning to Complete Bayes .....	<b>Error! Bookmark not defined.</b>
Practice 6.13.13 Challenge .....	<b>Error! Bookmark not defined.</b>
Practice 6.13.14.....	<b>Error! Bookmark not defined.</b>
7 : Naive Bayes Classifier.....	<b>Error! Bookmark not defined.</b>
7.1 Introduction .....	<b>Error! Bookmark not defined.</b>
7.2 Developing the Naïve Bayes Classifier .....	<b>Error! Bookmark not defined.</b>
Practice 7.2.1.....	<b>Error! Bookmark not defined.</b>
7.3 Trying out the Naïve Bayes Classifier .....	<b>Error! Bookmark not defined.</b>
7.4 Independence and Conditional Independence.....	<b>Error! Bookmark not defined.</b>
Practice 7.4.2.....	<b>Error! Bookmark not defined.</b>
7.5 Naïve Bayes example using the Golf data set.....	<b>Error! Bookmark not defined.</b>
Practice 7.5.3.....	<b>Error! Bookmark not defined.</b>
7.6 Naïve Bayes classification for class G.....	<b>Error! Bookmark not defined.</b>
Practice 7.6.4.....	<b>Error! Bookmark not defined.</b>
7.7 R implementation of the Naïve Bayes: Version 1 .....	<b>Error! Bookmark not defined.</b>

Practice 7.7.5.....	Error! Bookmark not defined.
7.8 Version 2 of Naïve Bayes in R.....	Error! Bookmark not defined.
Practice 7.8.6.....	Error! Bookmark not defined.
Practice 7.8.7.....	Error! Bookmark not defined.
Practice 7.8.8.....	Error! Bookmark not defined.
7.9 Conclusion.....	Error! Bookmark not defined.
8 Stored Model for Naive Bayes Classifier .....	Error! Bookmark not defined.
8.1 Introduction .....	Error! Bookmark not defined.
8.2 Building the Learned Model.....	Error! Bookmark not defined.
8.3 Matrices to hold the model parameters.....	Error! Bookmark not defined.
8.4 Building the Matrices .....	Error! Bookmark not defined.
8.5 Digression on the apply() functions. ....	Error! Bookmark not defined.
8.6 Avoiding Probabilities of Zero .....	Error! Bookmark not defined.
8.7 Other helper functions.....	Error! Bookmark not defined.
8.8 Split data into training and testing. ....	Error! Bookmark not defined.
Practice 8.8.1.....	Error! Bookmark not defined.
Practice 8.8.2.....	Error! Bookmark not defined.
Practice 8.8.3.....	Error! Bookmark not defined.
9 Review of Mathematics for Neural Networks .....	Error! Bookmark not defined.
9.1 Mathematical review of vectors .....	Error! Bookmark not defined.
9.2 Another perspective on $\mathbb{R}^2$ .....	Error! Bookmark not defined.
Practice 9.2.1.....	Error! Bookmark not defined.
9.3 Vectors as Arrows .....	Error! Bookmark not defined.
9.4 Dimension .....	Error! Bookmark not defined.
9.5 Operations on Vectors: Sum and Dot Product.....	Error! Bookmark not defined.
Practice 9.5.2.....	Error! Bookmark not defined.
9.6 Matrices .....	Error! Bookmark not defined.
9.7 Matrices as a vector of vectors .....	Error! Bookmark not defined.
9.8 Matrix Addition .....	Error! Bookmark not defined.
Practice 9.8.3.....	Error! Bookmark not defined.
9.9 Matrix Multiplication .....	Error! Bookmark not defined.
Practice 9.9.4.....	Error! Bookmark not defined.

9.10 Matrix Multiplication in $\mathbb{R}$ .....	<b>Error! Bookmark not defined.</b>
Practice 9.10.5.....	<b>Error! Bookmark not defined.</b>
9.11 Transpose of a matrix .....	<b>Error! Bookmark not defined.</b>
Practice 9.11.6.....	<b>Error! Bookmark not defined.</b>
Practice 9.11.7.....	<b>Error! Bookmark not defined.</b>
9.12 Functions of a single input variable .....	<b>Error! Bookmark not defined.</b>
9.13 Multivariate functions.....	<b>Error! Bookmark not defined.</b>
Practice 9.13.8.....	<b>Error! Bookmark not defined.</b>
9.14 Vector-valued functions.....	<b>Error! Bookmark not defined.</b>
9.15 Multivariate vector-valued functions .....	<b>Error! Bookmark not defined.</b>
9.16 Explicit versus implicit formula for a function .....	<b>Error! Bookmark not defined.</b>
Practice 9.16.9.....	<b>Error! Bookmark not defined.</b>
9.17 Single Variable Derivatives.....	<b>Error! Bookmark not defined.</b>
9.18 Sum and Difference Rules .....	<b>Error! Bookmark not defined.</b>
Sum Rule .....	<b>Error! Bookmark not defined.</b>
Practice 9.18.10.....	<b>Error! Bookmark not defined.</b>
9.19 Product Rule.....	<b>Error! Bookmark not defined.</b>
Practice 9.19.11.....	<b>Error! Bookmark not defined.</b>
9.20 Chain Rule .....	<b>Error! Bookmark not defined.</b>
9.21 Derivative of $e^x$ .....	<b>Error! Bookmark not defined.</b>
9.22 Sigmoid function and its derivative .....	<b>Error! Bookmark not defined.</b>
Practice 9.22.12.....	<b>Error! Bookmark not defined.</b>
9.23 Derivatives for Multivariate and Vector-valued Functions.....	<b>Error! Bookmark not defined.</b>
Practice 9.23.13.....	<b>Error! Bookmark not defined.</b>
9.24 Minimum and Maximums of a Function .....	<b>Error! Bookmark not defined.</b>
9.25 Partial Derivatives .....	<b>Error! Bookmark not defined.</b>
Practice 9.25.14.....	<b>Error! Bookmark not defined.</b>
Practice 9.25.15.....	<b>Error! Bookmark not defined.</b>
Practice 9.25.16.....	<b>Error! Bookmark not defined.</b>
9.26 Multivariate Version of the Chain Rule.....	<b>Error! Bookmark not defined.</b>
9.27 Example of Multivariate Chain Rule.....	<b>Error! Bookmark not defined.</b>
9.28 Without the Chain Rule.....	<b>Error! Bookmark not defined.</b>

9.29 Using the Chain Rule on our Example .....	<b>Error! Bookmark not defined.</b>
Practice 9.29.17.....	<b>Error! Bookmark not defined.</b>
Practice 9.29.18.....	<b>Error! Bookmark not defined.</b>
9.30 Gradient Descent with Partial Derivatives .....	<b>Error! Bookmark not defined.</b>
Practice 9.30.19.....	<b>Error! Bookmark not defined.</b>
9.31 Gradients Analytically .....	<b>Error! Bookmark not defined.</b>
Practice 9.31.20.....	<b>Error! Bookmark not defined.</b>
Practice 9.31.21.....	<b>Error! Bookmark not defined.</b>
9.32 Possible point of confusion -The gradient has the same number of elements as the number of <i>input</i> variables.....	<b>Error! Bookmark not defined.</b>
9.33 Notation .....	<b>Error! Bookmark not defined.</b>
Practice 9.33.22.....	<b>Error! Bookmark not defined.</b>
9.34 Conclusion.....	<b>Error! Bookmark not defined.</b>
10 Neural Networks -Feed Forward Process and Back Propagation Process	<b>Error! Bookmark not defined.</b>
10.1 1-1-1 Architecture Feed Forward Process .....	<b>Error! Bookmark not defined.</b>
10.2 Cost Function J .....	<b>Error! Bookmark not defined.</b>
Two Input Neural Network -Feed Forward.....	<b>Error! Bookmark not defined.</b>
10.3 2-3-1 Neural Network .....	<b>Error! Bookmark not defined.</b>
10.4 Weight Indices .....	<b>Error! Bookmark not defined.</b>
10.5 Hidden Layer Calculations.....	<b>Error! Bookmark not defined.</b>
10.6 Output Layer Calculations and Cost.....	<b>Error! Bookmark not defined.</b>
10.7 Calculation of $dJ/dw2$ .....	<b>Error! Bookmark not defined.</b>
Practice 10.7.1.....	<b>Error! Bookmark not defined.</b>
Practice 10.7.2.....	<b>Error! Bookmark not defined.</b>
10.8 Calculation of $dJ/dw1$ .....	<b>Error! Bookmark not defined.</b>
10.9 Updating weights and searching for the minimum cost.....	<b>Error! Bookmark not defined.</b>
Practice 10.9.3.....	<b>Error! Bookmark not defined.</b>
10.10 Backpropagation for the 2-1-1 Neural Network .....	<b>Error! Bookmark not defined.</b>
10.11 Define the matrices for the derivative and Delta2 .....	<b>Error! Bookmark not defined.</b>
10.12 Writing the update equations with matrices.....	<b>Error! Bookmark not defined.</b>
10.13 1-1-1 Neural Network R code .....	<b>Error! Bookmark not defined.</b>

Practice 10.13.4.....	Error! Bookmark not defined.
Practice 10.13.5.....	Error! Bookmark not defined.
Practice 10.13.6.....	Error! Bookmark not defined.
Practice 10.13.7.....	Error! Bookmark not defined.
Practice 10.13.8.....	Error! Bookmark not defined.
Practice 10.13.9.....	Error! Bookmark not defined.
10.14 Backpropagation for the 2-3-1 Neural Network in R .....	Error! Bookmark not defined.
Practice 10.14.10.....	Error! Bookmark not defined.
10.15 Derivatives for $W_2$ .....	Error! Bookmark not defined.
10.16 Derivatives for $W_1$ .....	Error! Bookmark not defined.
Practice 10.16.11 .....	Error! Bookmark not defined.
10.17 Updating the weights.....	Error! Bookmark not defined.
Practice 10.17.12.....	Error! Bookmark not defined.
10.18 Running an entire dataset through the 2-3-1 network.....	Error! Bookmark not defined.
Practice 10.18.13.....	Error! Bookmark not defined.
Practice 10.18.14.....	Error! Bookmark not defined.
Practice 10.18.15.....	Error! Bookmark not defined.
10.19 Repeated training: Epochs .....	Error! Bookmark not defined.
Practice 10.19.16.....	Error! Bookmark not defined.
10.20 Hyperparameters -Learning Rates and set.seed() Number of Nodes and Layers, Epochs .....	<b>Error! Bookmark not defined.</b>
10.21 Conclusion.....	Error! Bookmark not defined.
11 Programming a Neural Network using OOP in R .....	Error! Bookmark not defined.
11.1 Object Oriented Programming with R .....	Error! Bookmark not defined.
11.2 Classes as Blueprints .....	Error! Bookmark not defined.
11.3 Example of a Class.....	Error! Bookmark not defined.
11.4 Creating an Instance of the Class .....	Error! Bookmark not defined.
Practice 11.4.1.....	Error! Bookmark not defined.
11.5 A Neural Network Class.....	Error! Bookmark not defined.
Practice 11.5.2.....	Error! Bookmark not defined.
11.6 Why Use Classes for a Neural Network?.....	Error! Bookmark not defined.
11.7 Classes in R.....	Error! Bookmark not defined.

11.8 Fields .....	<b>Error! Bookmark not defined.</b>
Practice 11.8.3.....	<b>Error! Bookmark not defined.</b>
11.9 Methods .....	<b>Error! Bookmark not defined.</b>
11.10 Batch Updating .....	<b>Error! Bookmark not defined.</b>
Practice 11.10.4.....	<b>Error! Bookmark not defined.</b>
11.11 Preparation for Using the Neural Network.....	<b>Error! Bookmark not defined.</b>
11.12 Neural Network Class.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.5.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.6.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.7.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.8.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.9.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.10.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.11.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.12.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.13.....	<b>Error! Bookmark not defined.</b>
Practice 11.12.14.....	<b>Error! Bookmark not defined.</b>
11.13 Conclusion.....	<b>Error! Bookmark not defined.</b>
12 Adding in a Bias Term.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.1.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.2.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.3.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.4.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.5.....	<b>Error! Bookmark not defined.</b>
Practice 12.1.6.....	<b>Error! Bookmark not defined.</b>
12.2 backpropagation method begins here.....	<b>Error! Bookmark not defined.</b>
Practice 12.2.7.....	<b>Error! Bookmark not defined.</b>
Practice 12.2.8.....	<b>Error! Bookmark not defined.</b>
Practice 12.2.9.....	<b>Error! Bookmark not defined.</b>
12.3 JCost method begins here.....	<b>Error! Bookmark not defined.</b>
12.4 Steps for running the neural network.....	<b>Error! Bookmark not defined.</b>
13 Modular Version of Neural Networks for Deep Learning .....	<b>Error! Bookmark not defined.</b>

13.1 Back propagation .....	Error! Bookmark not defined.
Practice 13.1.1.....	Error! Bookmark not defined.
13.2 Generalized Layer Notation .....	Error! Bookmark not defined.
13.3 $\partial J \partial X = \partial J \partial Y$ ?.....	Error! Bookmark not defined.
13.4 Splitting Layers So They Represent Single Processes.....	Error! Bookmark not defined.
Practice 13.4.2.....	Error! Bookmark not defined.
13.5 Linear versus non-linear layers .....	Error! Bookmark not defined.
Practice 13.5.3.....	Error! Bookmark not defined.
Practice 13.5.4.....	Error! Bookmark not defined.
13.6 Linear fully connected layer.....	Error! Bookmark not defined.
13.7 Forward propagation for a linear fully connected layer .....	Error! Bookmark not defined.
13.8 Back propagation for a linear fully connected layer - Calculation of $\partial J \partial W, \partial J \partial B, \partial J \partial Y, \partial J \partial X$ .....	Error! Bookmark not defined.
Practice 13.8.5.....	Error! Bookmark not defined.
Practice 13.8.6.....	Error! Bookmark not defined.
Practice 13.8.7.....	Error! Bookmark not defined.
Practice 13.8.8.....	Error! Bookmark not defined.
13.9 Calculating $\partial J \partial B$ .....	Error! Bookmark not defined.
Practice 13.9.9.....	Error! Bookmark not defined.
13.10 Calculating $\partial J \partial X$ .....	Error! Bookmark not defined.
13.11 Coding a Fully Connected Layer .....	Error! Bookmark not defined.
Practice 13.11.10.....	Error! Bookmark not defined.
13.12 Activation Layer .....	Error! Bookmark not defined.
13.13 Forward Propagation for the Activation Layer .....	Error! Bookmark not defined.
13.14 Back Propagation for the Activation Layer .....	Error! Bookmark not defined.
Practice 13.14.11.....	Error! Bookmark not defined.
Practice 13.14.12.....	Error! Bookmark not defined.
13.15 Coding an Activation Layer .....	Error! Bookmark not defined.
13.16 Coding the Neural Network class, creating an instance, and running it.....	<b>Error! Bookmark not defined.</b>
14 Deep Learning with Convolutional Neural Networks .....	Error! Bookmark not defined.
14.1 Dataset -Handwritten Repository of Digits from 0 to 9 .....	Error! Bookmark not defined.

14.2 Converting images into inputs .....	<b>Error! Bookmark not defined.</b>
14.3 Convolution as a mathematical operation.....	<b>Error! Bookmark not defined.</b>
14.4 Forward Propagation for the Convolutional Layer .....	<b>Error! Bookmark not defined.</b>
Practice 14.4.1.....	<b>Error! Bookmark not defined.</b>
14.5 Convolution of Input and Filter Matrices.....	<b>Error! Bookmark not defined.</b>
Practice 14.5.2.....	<b>Error! Bookmark not defined.</b>
Practice 14.5.3.....	<b>Error! Bookmark not defined.</b>
Practice 14.5.4.....	<b>Error! Bookmark not defined.</b>
14.6 Further details in setting up a convolutional neural network .....	<b>Error! Bookmark not defined.</b>
14.7 Code for CNN .....	<b>Error! Bookmark not defined.</b>
14.8 Max Pooling Layer .....	<b>Error! Bookmark not defined.</b>
Practice 14.8.5.....	<b>Error! Bookmark not defined.</b>
Practice 14.8.6.....	<b>Error! Bookmark not defined.</b>
14.9 Flatten Layer .....	<b>Error! Bookmark not defined.</b>
Practice 14.9.7.....	<b>Error! Bookmark not defined.</b>
14.10 Network Class .....	<b>Error! Bookmark not defined.</b>
Practice 14.10.8.....	<b>Error! Bookmark not defined.</b>
Practice 14.10.9.....	<b>Error! Bookmark not defined.</b>
14.11 Getting the Data Ready.....	<b>Error! Bookmark not defined.</b>
Practice 14.11.10.....	<b>Error! Bookmark not defined.</b>
14.12 Building Neural Networks .....	<b>Error! Bookmark not defined.</b>
Practice 14.12.11.....	<b>Error! Bookmark not defined.</b>
Practice 14.12.12.....	<b>Error! Bookmark not defined.</b>
15 R Packages for Neural Networks, Deep Learning, and Naïve Bayes .....	<b>Error! Bookmark not defined.</b>
Practice 15.1.1.....	<b>Error! Bookmark not defined.</b>
15.2 R package: neuralnet .....	<b>Error! Bookmark not defined.</b>

# Table of Figures

Figure 1-1 Initial configuration of RStudio .....	18
Figure 1-2 Functions coded in chapter 1 .....	19
Figure 2-1 View of mtcars .....	<b>Error! Bookmark not defined.</b>
Figure 3-1 Examples of Venn Diagrams .....	<b>Error! Bookmark not defined.</b>
Figure 3-2 Missing data example .....	<b>Error! Bookmark not defined.</b>
Figure 3-3 RStudio view of missing data .....	<b>Error! Bookmark not defined.</b>
Figure 3-4 Functions we build or use in this chapter and their response to NAs and repetition of elements .....	<b>Error! Bookmark not defined.</b>
Figure 3-5 Universal Set and B as a subset of A .....	<b>Error! Bookmark not defined.</b>
Figure 3-6 Intersection of A and B .....	<b>Error! Bookmark not defined.</b>
Figure 3-7 Union of A and B .....	<b>Error! Bookmark not defined.</b>
Figure 3-8 Complement of A .....	<b>Error! Bookmark not defined.</b>
Figure 4-1 Roll a pair of dice .....	<b>Error! Bookmark not defined.</b>
Figure 4-2 Toss a coin.....	<b>Error! Bookmark not defined.</b>
Figure 4-3 Minimal deck of cards.....	<b>Error! Bookmark not defined.</b>
Figure 4-4 Standard 6-sided die .....	<b>Error! Bookmark not defined.</b>
Figure 4-5 Minimal deck of cards.....	<b>Error! Bookmark not defined.</b>
Figure 4-6 One observational unit can have many attributes .....	<b>Error! Bookmark not defined.</b>
Figure 4-7 All the instances have y for V1 and n for V2 .....	<b>Error! Bookmark not defined.</b>
Figure 4-8 View(A).....	<b>Error! Bookmark not defined.</b>
Figure 4-9 View(B).....	<b>Error! Bookmark not defined.</b>
Figure 4-10 View(C).....	<b>Error! Bookmark not defined.</b>
Figure 4-11 Since events can be thought of as sets, we will talk of the union of events ...	<b>Error! Bookmark not defined.</b>
Figure 4-12 If you try the code yourself, and scroll down, you will see that they are not all democrat. .....	<b>Error! Bookmark not defined.</b>
Figure 4-13 same as before.....	<b>Error! Bookmark not defined.</b>
Figure 4-14 Events can be thought of sets and so we will speak of the intersection of events .....	<b>Error! Bookmark not defined.</b>
Figure 4-15 The event of A not occurring as the complement of A.....	<b>Error! Bookmark not defined.</b>
Figure 4-16 G and H are not mutually exclusive. If a 5 is rolled we would say that both events have occurred. ....	<b>Error! Bookmark not defined.</b>
Figure 4-17 Disjoint sets.....	<b>Error! Bookmark not defined.</b>
Figure 4-18 Non-disjoint sets .....	<b>Error! Bookmark not defined.</b>
Figure 4-19 Event of getting a 1 or 3 or 5 when rolling a die .....	<b>Error! Bookmark not defined.</b>
Figure 4-20 Illustration of the events E and F .....	<b>Error! Bookmark not defined.</b>
Figure 4-21 RStudio view of the sample space for rolling a fair die .....	<b>Error! Bookmark not defined.</b>
Figure 4-22 Representation of tossing 2 fair coins .....	<b>Error! Bookmark not defined.</b>
Figure 4-23 RStudio view of representation of tossing 2 fair coins .....	<b>Error! Bookmark not defined.</b>
Figure 4-24 Graphic representation of the 3 events Red, Blue, and Yellow..	<b>Error! Bookmark not defined.</b>

Figure 4-25 Two boxes with blue and yellow balls. Since the box is selected at random each box has a 50-50 chance of being selected.....	<b>Error! Bookmark not defined.</b>
Figure 4-26 A partition of U into two sets .....	<b>Error! Bookmark not defined.</b>
Figure 4-27 A partition of U into four sets .....	<b>Error! Bookmark not defined.</b>
Figure 4-28 A partition of U into 4 sets and another subset of U called B ....	<b>Error! Bookmark not defined.</b>
Figure 4-29 B can be written as the disjoint union of the four sets indicated.....	<b>Error! Bookmark not defined.</b>
Figure 4-30 The event B written as the sum of two mutually exclusive events as indicated.....	<b>Error! Bookmark not defined.</b>
Figure 5-1 Functions coded in this chapter.....	<b>Error! Bookmark not defined.</b>
Figure 5-2 Three species of iris flowers .....	<b>Error! Bookmark not defined.</b>
Figure 5-3 In particular we measure the petal length and width and sepal length and width for each of the 150 specimens. Thus, we have 5 measurements on each specimen. ....	<b>Error! Bookmark not defined.</b>
Figure 5-4 In particular we measure the petal length and width and sepal length and width for each of the 150 specimens. Thus, we have 5 measurements on each specimen. ....	<b>Error! Bookmark not defined.</b>
Figure 5-5 Measurements of a sample of the dataset.....	<b>Error! Bookmark not defined.</b>
Figure 5-6 mtcars .....	<b>Error! Bookmark not defined.</b>
Figure 6-1 Functions coded in Chapter 6 .....	<b>Error! Bookmark not defined.</b>
Figure 6-2 Arthritis data set from the vcd package with attributes ID,Treatment, Sex,Age,Improved	<b>Error! Bookmark not defined.</b>
Figure 6-3 Some attributes have been removed .....	<b>Error! Bookmark not defined.</b>
Figure 7-1 Functions coded in Chapter 7 .....	<b>Error! Bookmark not defined.</b>
Figure 7-2 Golf data set.....	<b>Error! Bookmark not defined.</b>
Figure 7-3 Golf data set with just 2 input variables .....	<b>Error! Bookmark not defined.</b>
Figure 8-1 Functions coded in chapter 8 .....	<b>Error! Bookmark not defined.</b>
Figure 9-1 Tangent line (in blue) to the green curve .....	<b>Error! Bookmark not defined.</b>
Figure 9-2: Sigmoid curve as an example of general family of curves referred to as S-curves .....	<b>Error! Bookmark not defined.</b>
Figure 9-3 Average cost as a function of number of goods produced. We may want to find the lowest average cost .....	<b>Error! Bookmark not defined.</b>
Figure 9-4 A cost curve that has 2 local minima. Starting at B, we may only locate C rather than the absolute minimum at E .....	<b>Error! Bookmark not defined.</b>
Figure 9-5 Average cost as a function of number of products and cost of labor .....	<b>Error! Bookmark not defined.</b>
Figure 9-6 Intersection of paraboloid and vertical plane perpendicular to y-axis. The intersection of these two is a parabola.....	<b>Error! Bookmark not defined.</b>
Figure 9-7: The chain rule for $f \circ x$ where $f: \mathbb{R}^2 \rightarrow \mathbb{R}$ and $x: \mathbb{R}^1 \rightarrow \mathbb{R}^2$ ...	<b>Error! Bookmark not defined.</b>
Figure 9-8 The chain rule for $f \circ x$ where $f: \mathbb{R}^3 \rightarrow \mathbb{R}$ and $x: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ ....	<b>Error! Bookmark not defined.</b>
Figure 9-9 Paraboloid whose minimum is difficult to identify visually.....	<b>Error! Bookmark not defined.</b>
Figure 9-10 Various candidates for the gradient originating at the point A..	<b>Error! Bookmark not defined.</b>
Figure 10-1 Simplest neural network architecture: 1-1-1.....	<b>Error! Bookmark not defined.</b>
Figure 10-2 Two input node architecture: 2-1-1.....	<b>Error! Bookmark not defined.</b>
Figure 10-3 2-3-1 neural network architecture .....	<b>Error! Bookmark not defined.</b>
Figure 10-4 Functions coded in chapter 10 .....	<b>Error! Bookmark not defined.</b>

Figure 10-5 2-1-1 architecture and the cost function J.....	<b>Error! Bookmark not defined.</b>
Figure 10-6 2-3-1 architecture .....	<b>Error! Bookmark not defined.</b>
Figure 10-7 .....	<b>Error! Bookmark not defined.</b>
Figure 10-8 How to multiply rows and columns .....	<b>Error! Bookmark not defined.</b>
Figure 10-9 Visualization of the dimensions involved in the 2-3-1 architecture .....	<b>Error! Bookmark not defined.</b>
<b>defined.</b>	
Figure 10-10 Illustration of meaning of downstream and upstream.....	<b>Error! Bookmark not defined.</b>
Figure 10-11 .....	<b>Error! Bookmark not defined.</b>
Figure 10-12 .....	<b>Error! Bookmark not defined.</b>
Figure 10-13 Cost J as a function of a single weight w .....	<b>Error! Bookmark not defined.</b>
Figure 10-14 .....	<b>Error! Bookmark not defined.</b>
Figure 10-15 .....	<b>Error! Bookmark not defined.</b>
Figure 10-16 .....	<b>Error! Bookmark not defined.</b>
Figure 10-17 .....	<b>Error! Bookmark not defined.</b>
Figure 10-18 .....	<b>Error! Bookmark not defined.</b>
Figure 10-19 Plot of "mood" as a function of "calories_burned" and "hours_of_sleep" ...	<b>Error! Bookmark not defined.</b>
<b>not defined.</b>	
Figure 10-20 .....	<b>Error! Bookmark not defined.</b>
Figure 11-1 Functions coded in chapter 11 .....	<b>Error! Bookmark not defined.</b>
Figure 11-2 3 vectors representing 3 separate examples. The dimensions (x,y,z axes) are partial derivatives with respect to three weights .....	<b>Error! Bookmark not defined.</b>
Figure 11-3 Graph of ReLU activation function .....	<b>Error! Bookmark not defined.</b>
Figure 12-1 .....	<b>Error! Bookmark not defined.</b>
Figure 12-2 .....	<b>Error! Bookmark not defined.</b>
Figure 12-3 .....	<b>Error! Bookmark not defined.</b>
Figure 12-4 .....	<b>Error! Bookmark not defined.</b>
Figure 12-5 .....	<b>Error! Bookmark not defined.</b>
Figure 12-6 .....	<b>Error! Bookmark not defined.</b>
Figure 12-7 .....	<b>Error! Bookmark not defined.</b>
Figure 13-1 Functions coded in chapter 13 .....	<b>Error! Bookmark not defined.</b>
Figure 13-2: 1-1-1 Architecture. Note that the first layer has 2 X's. The first is the input and the second is the output. Of course, in the first node the input and output are the same and so we repeat X twice.	
.....	<b>Error! Bookmark not defined.</b>
Figure 13-3 Some calculations from layer 2 are used as an input when doing the calculations for layer 1 in this modular perspective to coding the neural network .....	<b>Error! Bookmark not defined.</b>
Figure 13-4 Information that will be provided to the code for layer 1.....	<b>Error! Bookmark not defined.</b>
Figure 13-5 Derivative calculation as local times downstream components. Note the multiplication sign in the arrow. Y is the output of this layer.....	<b>Error! Bookmark not defined.</b>
Figure 13-6.....	<b>Error! Bookmark not defined.</b>
Figure 13-7.....	<b>Error! Bookmark not defined.</b>
Figure 13-8 Each process is represented as a separate layer.....	<b>Error! Bookmark not defined.</b>
Figure 13-9: General layer behavior .....	<b>Error! Bookmark not defined.</b>
Figure 13-10: Forward propagation .....	<b>Error! Bookmark not defined.</b>
Figure 13-11 Single Process Fully Connected Linear Layer .....	<b>Error! Bookmark not defined.</b>

Figure 13-12 Activation Layer..... **Error! Bookmark not defined.**  
 Figure 14-1 Functions coded in chapter 14 ..... **Error! Bookmark not defined.**  
 Figure 14-2 Sample images from MNIST test dataset ..... **Error! Bookmark not defined.**  
 Figure 14-3 8 x 8 array of gray scale pixel (picture elements). Each pixel is a shade of gray on a scale of 0 to 255 ..... **Error! Bookmark not defined.**  
 Figure 14-4 Color low resolution image where the pixel nature is evident .. **Error! Bookmark not defined.**  
 Figure 14-5 Input Matrix ..... **Error! Bookmark not defined.**  
 Figure 14-6 Filter Matrix ..... **Error! Bookmark not defined.**  
 Figure 14-7 Correlation of image matrix and filter matrix to output the correlation feature matrix .... **Error! Bookmark not defined.**  
**Bookmark not defined.**  
 Figure 14-8 Convolution of Input Matrix and Filter Matrix ..... **Error! Bookmark not defined.**  
 Figure 14-9 Clockwise rotation of yellow partials matrix of E with respect to O around green filter matrix will generate the equations partial derivative of E with respect to X. This is a visual representation of the operation known as Full Convolution. Only three positions are shown but if the rotation continues there are 9 positions, generating the 9 equations. .... **Error! Bookmark not defined.**  
 Figure 14-10: 2x2 Max-pooling. The underlined values are the largest in their blocks. Thus, they are the ones that appear in the new layer. Note that in this diagram the filter itself is not shown. However, the filter is of the same size as the output..... **Error! Bookmark not defined.**  
 Figure 14-11 Mask records from which cells the maximums were obtained.**Error! Bookmark not defined.**  
 Figure 14-12 The process of flattening as applied to a single matrix. .... **Error! Bookmark not defined.**  
 Figure 14-13 The process of flattening when applied to a volume consisting of more than one matrix. .... **Error! Bookmark not defined.**  
 Figure 14-14 y\_train contains the labels for the elements of x\_train. .... **Error! Bookmark not defined.**

original variable	newVar1	newVar2	newVar3	newVar4	newVar5	newVar6	newVar7	newVar8	newVar9	newVar10
5	0	0	0	0	0	1	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1

Figure 14-15 Transform the representation of a variable the originally has 10 levels to 10 new binary variables. This is called one-hot encoding or creating dummy variables. .... **Error! Bookmark not defined.**  
 Figure 15-1 ..... **Error! Bookmark not defined.**  
 Figure 15-2 ..... **Error! Bookmark not defined.**

# 1 -Functions Tutorial

- Functions with and without arguments
- Defining versus running
- Arguments
- Parameter versus Argument
- Argument Order
- Environments
- Scope

*Replicating the results shown here*

Note that this text uses the following version of R: version 3.4.2 (2017-09-28), To obtain the same results as this text you may install this version at CRAN.

*Knowledge of R*

It is assumed that the reader has gone through the tutorial offered through an R package called Swirl. You can find out more information at <https://swirlstats.com/>. The tutorial can be run inside RStudio and gives a very good course in R. It is expected that the reader has completed the tutorial through the section on functions.

According to the website you can run the tutorial in RStudio by following steps below.

Step 1: Get R. In order to run swirl, you must have R 3.1.0 or later installed on your computer. ...

Step 2 (recommended): Get RStudio. ...

Step 3: Install swirl. ...

Step 4: Start swirl. ...

Step 5: Install an interactive course. ...

Step 6: Have fun!

## Getting set up

To get things going you should install R and install the free version of RStudio. Here are the steps to follow.

1. Download and install R from <https://cran.r-project.org/>. On Windows, if you don't mind living dangerously, I recommend that you use the "install as administrator" option because sometimes R wants to create folders, for example when new packages are installed, and if it is not installed and run as administrator, Windows may block the creation of those folders.
2. Download and install the free version of RStudio from <https://www.rstudio.com>. Same remarks as above regarding "install as administrator".
3. Run RStudio
4. Configure RStudio to look the way you want. You can adjust the appearance in the Global Options menu which is under Tools. When you open RStudio it may look something like this.

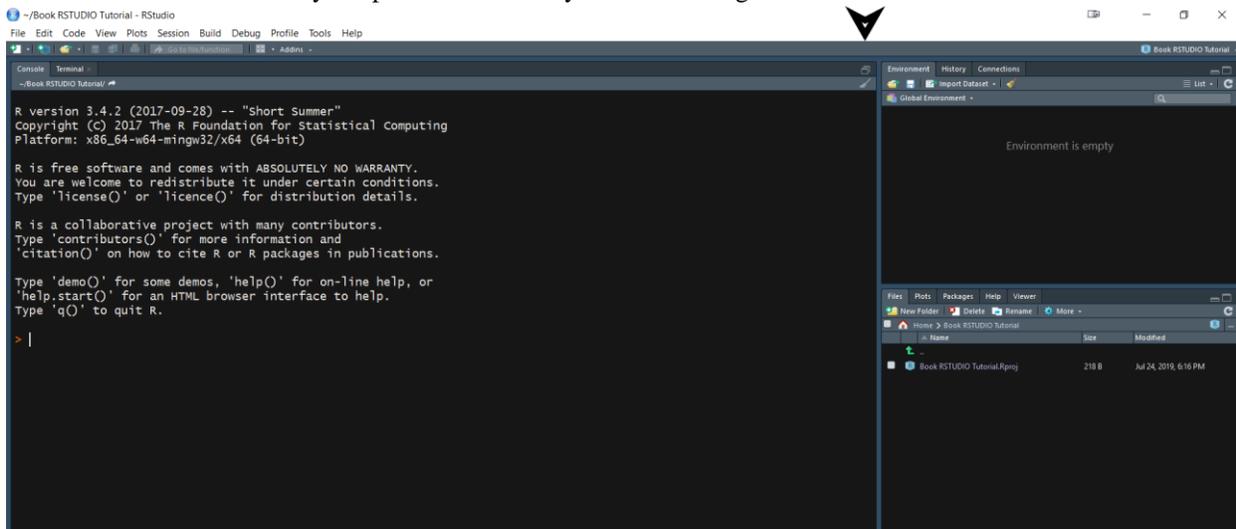


Figure 1-1 Initial configuration of RStudio

5. The so-called console window is the large window on the left. On the right are two other windows. Each window has various tabs that you can select. The first tab on the top right-hand window is the Environment tab and it will show any variables you have created, functions you have created, datasets that have been loaded and so on. The bottom window currently is showing the first tab which is just a file browser for the current directory that you are working in. You can see that the current directory I have open is Home>Book RSTUDIO Tutorial. There is one window which is not showing now that is important to have. It is called the source window. The console allows you to type single commands into R and when you hit the enter key, R will run that code. However, most of the time, when I am writing code, I don't want to evaluate each time I write a new line of code. I want to be able to write several lines of code first and then run it. This can't be done in the console window. The source window is used for this. To show the source window you can either click on the icon of two little boxes where the arrow is in the figure above or go to Tools<Global Options<Pane Layout. I prefer to have the console window above and the source below, but you can arrange as you wish. Also, as you can see, my background is black. This can also be adjusted in Global Options, as can many other options. Once you have the source window visible you can type code in it and then if you want to run a particular set of lines, select those lines, and then press the button on the upper right-hand side of the source window that says Run. Instead of pressing the button you can also select the code to be run and press the control key (on Windows) and simultaneously press the enter key.

To run the tutorial, you need to install the Swirl package. The simplest way to install a package is to go to Tools>Install Packages. Then type in the name of the package you are looking for, in this case swirl. Then click the install button. RStudio will download and install the package and when it is finished you will see the prompt symbol (looks like '>') in the console. According to the website on Swirl, next type `library("swirl")` and hit the enter key. Then type `swirl()` and hit the enter key. Next according to the website it says, "the first time you start swirl, you'll be prompted to install a course. You can either install one of the recommended courses or visit our course repository for more options. There are even more courses available from the Swirl Course Network. If you

follow the instructions it will bring you to a screen where you can choose the course you want to study". The recommended course for this text is called [R Programming: The basics of programming in R](#) and it is expected that you have completed up to and including the section on functions.

### Functions Tutorial

It is assumed that you have gone through the above tutorial at this point. Some of the following is a review of the Swirl course.

Our tutorial will concentrate on building functions since this is a critical skill in the material that follows.

Functions coded in this chapter
myFirstFunction
addFive
addThem
divid
outerFunction
embeddedFunction

Figure 1-2 Functions coded in chapter 1

## 1.2 Functions

If you have used R at all you have already worked with many functions. A function we often encounter is

```
mean(x)
```

```
x=1:5
```

```
mean(x)
```

```
## [1] 3
```

Another easy function is the function to get the date on the computer system you are using, `Sys.Date()`.

```
Sys.Date()
```

```
## [1] "2018-10-02"
```

There is an important difference between these two functions. They both have names and they both have parentheses but there is nothing inside the parentheses when we run `Sys.Date()`, however there is something inside the parentheses when we run `mean(x)`. The `mean()` function requires an input. In fact, if we run it without one, we get an error message.

```
mean()
```

```
Error in mean.default() : argument "x" is missing, with no default
```

Admittedly, R has error messages that are not so easy to figure out but if you skim them you can sometimes figure out what kind of problem has occurred.

### Write our first R function

Let's make this simple. To write a function is as simple as

```
myFirstFunction <- function(){  
  print("Hi There!")  
}
```

This shows the syntax of creating a function. We can divide the construction of this function into 3 parts. Before reading what these 3 parts are, look at the above statement and make a guess as to where the 3 parts might be (I am not referring to the <- symbol as one of the parts but of course that needs to be there).

1. It needs a name<sup>1</sup>:

```
myFirstFunction
```

2. You need to tell R that this name is the name of a function (as opposed to a text (character) variable, or a vector, or a list, or whatever else it might be).

```
function()
```

3. You want the function to do something and you put that inside the curly brackets {}

```
{  
  print("Hi There!")  
}
```

This function will just print out a string: "Hi There!". Note that what is written between the curly brackets is referred to as the body of the function. The body of our function contains

```
print("Hi There!")
```

## Defining the function versus running it

Although we define a function, nothing happens until we run it. That is done by typing its name along with the succeeding parentheses.

```
myFirstFunction()
```

Note that in RStudio, if the definition is typed into the console and the enter key is hit then the function is defined. However, if the definition is typed into the Source panel, it won't be defined until the *code is run* in the Source panel.

However, even after the function is defined, there will be no response from R until the function is run (this is usually referred to as "calling the function"). So, Hi There! will not appear in the console until myFirstFunction() is run, either by typing myFirstFunction() into the console and hitting enter, or typing myFirstFunction() into the Source

---

<sup>1</sup> There are cases where it is not necessary to name a function. That topic is called anonymous functions. We will not discuss it here.

panel and running it. If either of these is performed, the function `myFirstFunction()` will be called, and then it will perform the code within the curly brackets, which in this case is to print out Hi There!

Functions can do far more than simply print a string, but the code above is the full implementation of a function.

## Practice 1.2.2

1. Write a function named *fact* that will print the string “Tokyo is in Japan”.

Hopefully you will have functions do more than just print a string but nevertheless we have constructed a basic function.

What if you wanted to write a function to print the string "Tokyo is in Japan", how would you write that? Try it. And how would you run it (call it)? Try this in your Rstudio. If you can do this, you have created your own function.

### Arguments

Now try something a little more interesting. Write a function to add 5 to a given input. In this case, we need to add one more piece to the syntax of writing a function. The function needs to accept an input. It needs to accept given number. We say the function will have an argument. The etymology of the word argument is fairly convoluted and comes from something in Old English. But for us, argument refers to the input that we put into the function. If we call the function `addFive()` we want it to take an input, say 7, and then output 7+5.

Ok, so now we have decided what we want `addFive()` to do. Now we have to write it. It needs the first 3 parts mentioned above plus a place to put the argument. Also of course we have to change what is put into the curly brackets `{ }` because we don't want the function to print out Hi There. We want it to add 5 to the input.

```
addFive <- function(inp){  
  outp=inp+5  
  return(outp)  
}
```

We need a symbol to hold the input. We have chosen `inp`. Then inside the curly brackets we add 5 to `inp` and assign that to the variable `outp`. Finally we use the function `return()`. `return()` at the end of our code will determine what the output of the function will be. In our case the output is contained in `outp`. If we run the function with an argument of 7 we should expect an output of 12.

```
addFive(7)  
## [1] 12
```

## 1.3 Parameter versus argument

If you want to be able to read explanations on the web about programming issues you need to understand the difference between the two words *parameter* and *argument*. We have used the term argument. What is meant by parameter? We want to distinguish between what the user puts into our function (in the last case it was 7) versus the symbol that holds that value, and which is later used in the body of the function. The symbol we used to hold the argument is `inp`. This would be called a parameter. Thus, the argument entered into the function is 7 but the parameter is `inp`. It is very useful to be able to distinguish between these two when discussing our programming code and so the concept should be well understood.

We could certainly write a function which accepts 2 arguments. In that case we would use two parameters. For example if we want to take two inputs and get their sum we could write a function called `addThem()`. We hope that if we put 1 and 3 into `addThem()` it will return or output 4.

```
addThem <- function(inp1,inp2){
  outp=inp1 + inp2
  return(outp)
}
addThem(1,3)
```

### Practice 1.3.3

1. What are the parameters in addThem()?
2. What were the arguments we used in the addThem() test above?
3. What was returned in the addThem() test above?

In the addThem() function we said there were two parameters inp1 and inp2. We will refer to the list of all the parameters in the definition of the function as the parameter list. So here we would say that the parameter list has two elements inp1 and inp2. A function could have any number of parameters in the parameter list. One way to get an idea of what a function is doing is to look at the body of the function (what is inside the curly brackets). If there is a variable inside the curly brackets, then check to see if that is in the parameter list. If it is, it means that this value will be supplied when the function is called and then used in the way that is written inside the curly brackets. Another good practice for understanding what a function does, is to check the output of the function. That is, check the final return() statement of the function. This will tell you what is output and the form of the output. So far, we have had single numbers as the output of our functions, but it is possible to output other types like lists or vectors<sup>2</sup>. Checking the return statement is a quick way to get an idea about the function. Note also that if the function does not have a return statement, it will not show the value of the output when you run the function but it is still assignable to a variable which when printed (or just typed at the command line), will show that value.

```
addThem <- function(inp1,inp2){
  outp=inp1 + inp2
  #Leave out the return statement
}
addThem(1,3)
k=addThem(1,3)
k
## [1] 4
```

Some readers may object to the sentence above where we wrote, “So far, we have had single numbers as the output of our functions”, saying we output the text string “Hi There!”. In programming we make a distinction between what a function prints out and what it returns. It seems like an odd distinction to make and actually in R it may not be an important distinction but typically in most programming languages we cannot assign the value of the function myFirstFunction() to another variable, whereas we can certainly assign the output of addThem(1,3) to a variable. However, in R it is actually possible to assign the myFirstFunction() to a variable. This is not typical for most programming languages.

```
y=addThem(1,3)
y
## [1] 4
y+3
## [1] 7
```

---

<sup>2</sup> Outputting a list is a convenient way to output more than one item.

*#This actually works in R. Notice that even during the assignment, the function actually runs and actually outputs "Hi There!".*

```
x=myFirstFunction()  
## [1] "Hi There!"  
  
X  
  
x  
  
## [1] "Hi There!"
```

## 1.4 Argument Order and Parameter Names

In the calling of a function (when you actually use it) when you explicitly designate argument values by name using the corresponding parameter (in our last example, if you wrote `inp2=1,input1=3`), the ordering of the arguments becomes unimportant

Let's create a function called `divid()` which will take 2 arguments and divide the first by the second.

```
divid <- function(numerator,denominator){  
  result=numerator/denominator  
  return(result)  
}  
  
#The order of the arguments clearly matters.  
divid(8,2)  
  
## [1] 4  
  
divid(2,8)  
  
## [1] 0.25
```

*#If we specify which parameter, we want a particular argument to be assigned to, then the order does not matter. Here we are putting 2 first and then 8 but we have specified that the 2 is for the denominator parameter not the numerator parameter and the 8 is for the numerator parameter.*

```
divid(denominator = 2, numerator = 8)  
## [1] 4
```

### Practice 1.4.4

1. Write a function called *poly*, with 2 arguments which adds the square of one of them to the cube of the second.
2. Run `poly(2,3)` then run `poly(3,2)`
3. Try switching the order of the arguments but using the parameter names `poly(sq=2,cu=3)` and `poly(cu=3,sq=2)`. What is your conclusion?
4. What happens if you run `poly(cu=3,2)`?

Before we go on, we briefly discuss the concept of environment in R.

## 1.5 Environments

When we start up R and then create a variable, for example

```
x <- 7
```

then R needs to keep a record of the fact that a variable `x` has been created and that variable has been assigned the value 7. We say that the variable `x` has been stored in the global environment. An environment is just a place to store variables. It is like a piece of paper with the names of each variable and the values of those variables. It also keeps a record of the functions that have been created. In fact, there is a command we can run to see what is currently on that “sheet of paper”, i.e., what is in the global environment. It looks like this.

```
ls(globalenv())
```

The `ls()` function will give a list of items and so when we put `globalenv()` into it, we find out what is in

```
globalenv() .
```

For example, if we created the function called `myFirstFunction()` and also created the variable `x`, then we would get

```
myFirstFunction <- function(){
  print("Hi There!")
}
myFirstFunction()

## [1] "Hi There!"

x <- 7
ls(globalenv())

## [1] "myFirstFunction" "x"
```

This tells us that `myFirstFunction` and `x` are in the global environment.

We can actually create separate environments. This is done with

```
my.env <- new.env()
#Now we have created a new environment. It is currently empty
ls(my.env)

## character(0)

#However, we can assign variables values in this new (separate) environment like this.
assign("y", 100, envir=my.env)
#We can also use the $ sign to assign variables to the new environment, like this.
my.env$thursday = "The weather is sunny"
#Now if we check the contents of the my.env we will see both of these variables.
ls(my.env)

## [1] "thursday" "y"
```

*#So each environment is like a separate sheet of paper and they contain all the variable definitions (and function names as well).*

*#In fact, we can have a variable x in the global environment and another, separate variable, x, in the my.env environment. They can have different values even though they both are called x.*

```
assign("x", 20, envir=my.env)
```

*#We can use the name of the environment, followed by a \$ sign, followed by the name of the variable to see what the values of the variables are. (Note, the name of the global environment is .Global)*

```
.GlobalEnv$x
```

```
## [1] 7
```

```
my.env$x
```

```
## [1] 20
```

Generally, we don't create new environments for most of our programming tasks, but it is useful to see how once created, the items in the environment can be accessed. On the other hand, R will create a new environment every time any function is called. In our addthem() example

```
addThem <- function(inp1,inp2){  
  outp=inp1 + inp2  
  
}
```

if we run addThem(3,2), then a new environment is created (a new sheet of paper) which will contain the variables inp1, inp2, and outp along with the values each variable is assigned.

By the way, a phrase often used in computer science is "bindings". This usually refers to the assignment of a value to a variable. Thus we could say that when we call addThem(3,2), R creates a new environment with 3 bindings<sup>3</sup>.

We will see in a moment, that the concept of environments is tied to the important concept of scoping.

We saw above that we can have a variable named x in two separate environments and even though both variables are called x, they don't have the same values. That is, the fact that two variables have the same name is akin to two people having the same name. They may have the same name, but they certainly are not the same people. With regard to our 2 variables, if they reside in different environments, they are different variables. Now we said earlier, that when a function is called, it creates its own environment, its own piece of paper, where its variables and values are stored (where its bindings are stored).

Next, we consider what happens if we define a function inside another function. What do the R rules say about variables defined in the outer function. What do R rules say about variables defined in the inner function? If a variable is defined in the outer function, will R allow the inner function to access that variable? And visa-versa, if a variable is defined in the inner function, can the outer function access it? And also, if a variable is defined in a function, can we access that variable outside the function. These are often described as properties of the scope of a variable.

What is the scope of a variable defined within a function? Does it extend to functions that are defined within this function? Does the scope extend outside the function? Suppose we have a function called outerFunction(), and then

---

<sup>3</sup> More generally, binding refers to a mapping of one item to another.

inside that function we define another function called `embeddedFunction()` -yes, we are allowed to define functions inside other functions. What does the notion of environment imply about this situation?

```
#We define a function within another function. Then we call the outer function. The outer function also calls the embedded function (by way of a print statement)
outerFunction<- function(){
  myVar=2
  embeddedFunction <- function(){

    myVar=5
    return(myVar)
  }

  print(embeddedFunction())
  return(myVar)
}
```

What will happen when we call `outerFunction()`? In particular, when it returns `myVar`, which `myVar` will it return, the `myVar` defined in `outerFunction` (and thus return a 2) or the `myVar` defined in the `embeddedfunction` (and thus return a 5). Further, what will `print(embeddedFunction)` do? These are decisions that the designers of R had to make. Lets see what they decided R should do.

```
#Call outer function, which will return its myVar value but also print the output of the embedded function
outerFunction()
```

```
## [1] 5
```

```
## [1] 2
```

What is the explanation for the above *output* of 5 and then 2? Note that at the end of the definition of `outerFunction()` we have both a `print()` statement and a `return()` statement. This is why there are 2 lines of output when we run `outerFunction()`. Why are the outputs not the same? They are both returning (or printing) `myVar`. `embeddedFunction()` returns `myVar` and so does `outerFunction()`, so why don't they both return the same value?

First notice that the first line of the body of `outerFunction()` defines a variable called `myVar`. Next in the body of `outerFunction()` there is a new function being defined called `embeddedFunction()`. That is, `embeddedFunction()` is being defined within `outerFunction()`. Now examine

the body of `embeddedFunction()`. It also defines a variable called `myVar`. Are these two variables the same or do they just have the same name.

Remember that when a function is called it creates its own piece of paper to write its variables and values. At this point in the code, we have not called either function yet. But after defining the functions above, we do call the function `outerFunction()`. When `outerFunction()` runs, it will call `embeddedFunction()`, within the `print()` statement. When `print(embeddedFunction())` is run, this causes `embeddedFunction()` to be called and a new environment is created where the variable `myVar` is written with a value of 5. At the end of the execution of `embeddedFunction()`, the return statement with `myVar` will give 5. However, after `embeddedFunction()` returns 5, it completes its work. We say that then “control” passes back to `outerFunction()`. This will mean that we are switching to the environment of `outerFunction()`. When, finally, the last statement, `return(myVar)`, in `outerFunction()` is run, since this occurs within the environment of `outerFunction()` `myVar` will be 2.

There is one more concept associated with environments that we should discuss. Environments are said to be “nested”. That is to say, in our example, if `myVar` was only defined in `outerFunction()` and not defined within `embeddedFunction()`, then we might expect an error to occur when

we run our code since `embeddedFunction()` will run and try to return `myVar` but `myVar` does not exist within `embeddedFunction()`'s environment.

However, that is not what happens. What R does is search for `myVar` in other environments. In particular, R searches in the calling function's environment<sup>4</sup>. The calling function in this case is `outerFunction()` and R will find a variable called `myVar` there and this is the variable that will be used in . In this sense the environments are nested. Here is the example where we do not define `myVar` within the embedded function but try to access it within the embedded function anyway.

```
outerFunction<- function(){
  myVar=2
  embeddedFunction <- function(){
    #In this example, we do not create a variable called myVar within the emb
edded function.
    #myVar=5
    return(myVar)
  }

  print(embeddedFunction())
  return(myVar)
}
#Call outer function, which will return its myVar value but also print the ou
tput of the embedded function
outerFunction()

## [1] 2
## [1] 2
```

---

<sup>4</sup> Actually, R searches even beyond the calling functions environment. To see all the environments that R will search in (the so-called search path), run the command `search()`.

We see that the value of `myVar` is 2 in both cases, but in particular it is 2 in the embedded function even though it was not defined there. R finds `myVar` in the calling function `outerFunction()` and uses that binding.

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_1^{(2)}} \\ \frac{\partial J}{\partial w_2^{(2)}} \\ \frac{\partial J}{\partial w_3^{(2)}} \end{bmatrix} \quad (1.1)$$