CWRU DSCI351-451: Exploratory Data Science

Roger H. French, JiQi Liu

29 November, 2018

Contents

14.2.1.1 Reading, Homeworks, Projects, SemProjects . . . 1
14.2.1.2 Syllabus . . . 2
14.2.1.3 Final Exam (worth 20 pts) . . . 2
14.2.1.3.1 Final Exam Format . . . 2
14.2.1.3.2 Types of Questions . . . 2
14.2.1.3.3 Points per question . . . 2
14.2.1.4 Course Evaluations . . . 2
14.2.1.5 Questions on Course . . . 2
14.2.1.5.1 Overarching Goal of Course . . . 4
14.2.1.5.2 Utility of the 3 text books (R4DS, OIS, ISLR) . . . 4
14.2.1.5.3 The 3 books we used . . . 4
14.2.1.5.4 Git Class Repo structure to class . . . 4
14.2.1.6 Some CWRU alums in Computing . . . 4
14.2.1.6.1 Bill Gropp: National Center for Supercomputing Applications(NCSA) 4
14.2.1.6.2 Donald Knuth: TeX, The Art of Computer Programming . . . 4
14.2.1.6.3 Peter Tippett: Norton Antivirus etc. . . . 4
14.2.1.7 Privacy, Openness, Security, Ethics. Value . . . 4
14.2.1.8 A quick introduction to machine learning in R with caret . . . 5
14.2.1.8.1 What is machine learning? . . . 5
14.2.1.8.2 A simple example . . . 5
14.2.1.8.3 How much math you really need to know . . . 7
14.2.1.8.4 A quick introduction to caret . . . 8
14.2.1.8.5 Caret simplifies machine learning in R . . . 8
14.2.1.8.6 Caret's syntax . . . 9
14.2.1.8.7 The train() function . . . 9
14.2.1.8.8 Formula notation . . . 11
14.2.1.8.9 The data = parameter . . . 12
14.2.1.8.10 The method = parameter . . . 12
14.2.1.8.11 Next steps . . . 13
14.2.1.9 The Perceptron (Neural Networks) . . . 13
14.2.1.10 Links for Information in This Document . . . 14
14.2.1.10.1 R Machine Learning Essentials, by Michele Usuelli, Packt Publishing, 2014 . . . 14
14.2.1.10.2 A quick introduction to machine learning in R with caret . . . 14

14.2.1.1 Reading, Homeworks, Projects, SemProjects

- Homework:
  –
- Readings:
  –
- Projects:
  –
- 451 SemProjects:

- Report Outs 3 next week 15a 15b
- Final Report Due By Midnight, TUESDAY Dec. 18th 2017
- Format, sectioning, citations to packages and literature all defined
- Turn In Rmd, datafiles, tidy data, databook, codebook, scripts and report
- Rmd and R Script files must have authorship, versioning, license terms

14.2.1.2 Syllabus

14.2.1.3 Final Exam (worth 20 pts)

- Will be held Monday 12/17/2017
- From 12pm to 3pm
- In Olin 313, Students may need to bring their computers
- Comprehensive overview of the course

14.2.1.3.1 Final Exam Format

- The exam will appear in the prof repo
- In /assignments/finalexam folder
- Done as Rmd file to turn in as pdf report
- Submit Final Exam pdf to the Canvas Assignment Page

14.2.1.3.2 Types of Questions

- 6 questions total
- OI Stats questions to do
- Data Analysis: Tidying, EDA, Linear Regression
- 5 Paragraph Essay Question with cites: about Data Science
  - Citations to literature supporting your discussion

14.2.1.3.3 Points per question

- 1. OIS 2 pts
- 2. OIS 2 pts
- 3. EDA 2 pts
- 4. Essay 4 pts
- 5. EDA on Real Dataset problem 5 pts
- 6. Linear Regression on a dataset 5 pts

14.2.1.4 Course Evaluations

- Please fill out and give feedback
  - On what works, what needs improvement
- Course Eval Form To Fill Out

14.2.1.5 Questions on Course

Day:Date	Foundation	Practicum	Reading	Due
w1a:Tu:8/28/18	ODS Tool Chain	R, Rstudio, Git		
w1b:Th:8/30/18	Setup ODS Tool Chain	Bash, Git, Twitter	PRP4-33	HW1
w2a:Tu:9/4/18	What is Data Science	OIS:Intro2R	PRP35-64	**HW1 Due**
w2b:Th:9/6/18	Data Analytic Style, Git	451SempProj, Git	PRP65-93, OI1-1.9	HW2
w3a:Tu:9/11/18*	Struct. of Data Analysis	ISLR:Intro2R, Loops	PRP94-116, OIS3	**HW2 Due**
w3b:Th:9/13/18*	OIS3 Intro to Data	GapMinder, Dplyr, Magrittr		
w4a:Tu:9/18/18	OIS3, Intro2Data part 2, Data	EDA: PET Degr.	EDA1-31	Proj1
w4b:Th:9/20/18	Hypothesis Testing	GGPlot2 Tutorial	EDA32-58	HW3
w5a:Tu:9/25/18	Distributions	SemProj RepOut1	R4DS1-3	**HW3 Due**
w5b:Th:9/27/18	Wickham DSCI in Tidyverse	SemProj RepOut1	R4DS4-6	**SemProj1,**
w6a:Tu:10/2/18	OIS Found. of Inference	Inference	R4DS7-8	**Proj1 Due**
w6b:Th:10/4/18		Midterm Review	R4DS9-16 Wrangle	
w7a:Tu:10/9/18*	Summ. Stats & Vis.	Data Wrangling		
w7b:Th:10/11/18*	**MIDTERM EXAM**			HW4
w8a:Tu:10/16/18	Numerical Inference	Tidy Check Explore	OIS4	**HW4 Due**
w8b:Th:10/18/18	Algorithms, Models	Pairwise Corr. Plots	OIS5.1-4	Proj 2, HW5
Tu:10/23	**CWRU FALL BREAK**		R4DS17-21 Program	
w9b:Th:10/25/18	Categorical Infer	Predictive Analytics	OIS6.1,2	
w10a:Tu:10/30/18	SemProj	SemProj	OIS7	**SemProj2 HW5 Du**
w10b:Th:11/1/18	Lin. Regr.	Lin. Regr.	OIS8	**Proj.2 due**
w11a:Tu:11/6/18	Inf. for Regression	Curse of Dim.	OIS8	Proj 3
w11b:Th:11/8/18	Model Accuracy	Training Testing	ISLR3	HW6
w12a:Tu:11/13/18	Multiple Regr.	Mul. Regr. & Pred.	ISLR4	**HW6 due**
w12b:Th:11/15/18	Classification		ISLR6	
w13a:Tu:11/20/18	Classification	Clustering	ISLR5	**Proj 3 due**
Th:11/22/18	**THANKSGIVING**			Proj 4
w14a:Tu:11/27/18	Big Data	Hadoop		
w14b:Th:11/29/18	InfoSec	VerisDB		**SemProj3**
w15a:Tu:12/4/18	SemProj ReportOut3			
w15b:Th:12/6/18	SemProj ReportOut3			**Proj4**
	FINAL EXAM	**Monday12/17, 12:00-3:00pm**	Olin 313	**SemProj4 due**

Figure 1: DSCI351/451 Syllabus

14.2.1.5.1 Overarching Goal of Course

- Teach you how to do real data analysis projects
  - Using a modern data analysis tool chain
  - Using real-world and lab-based (messy) datasets
- Learn EDA to explore and discover insights from your data
  - And identify new data and metadata needed for data assembly

To achieve these goals

- What could be done better

14.2.1.5.2 Utility of the 3 text books (R4DS, OIS, ISLR)

- Which did you find useful,
- Which were not useful

14.2.1.5.3 The 3 books we used

- (R4DS) R for Data Science
- (OIS) Open Intro Stats v3
- (ISLR) Introduction to Statistical Learning with Applications in R

14.2.1.5.4 Git Class Repo structure to class

- This is a basic open-source collaboration method
  - did not use repo for turning in assignments
  - better by Git or by Blackboard/Canvas?

14.2.1.6 Some CWRU alums in Computing

14.2.1.6.1 Bill Gropp: National Center for Supercomputing Applications(NCSA)

14.2.1.6.2 Donald Knuth: TeX, The Art of Computer Programming

14.2.1.6.3 Peter Tippett: Norton Antivirus etc.

Things he has done

- History & Development of Norton AntiVirus
- Verizon Data Breach Investigation Report
  - 2018 DBIR
- Veris: The Vocabulary for Event Recording and Incident Sharing
  - Veris DB, an open source database of data breaches

14.2.1.7 Privacy, Openness, Security, Ethics. Value

These are important concepts in Data Science

- We'll look at Security and Data Breaches Today

14.2.1.8 A quick introduction to machine learning in R with caret

If you've been using R for a while,

- and you've been working with
  - basic data visualization and data exploration techniques,
- the next logical step is to start learning some machine learning.

To help you begin learning about machine learning in R,

- lets introduce you to an R package: the caret package.

We'll build a very simple machine learning model

- as a way to learn some of caret's basic syntax and functionality.

But before diving into caret,

- let's quickly discuss what machine learning is
  - and why we use it.

14.2.1.8.1 What is machine learning?

Machine learning is

- the study of data-driven, computational methods
- for making inferences and predictions.

Without going into extreme depth here,

- let's unpack that by looking at an example.

14.2.1.8.2 A simple example

Imagine that you want to understand

- the relationship between car weight and car fuel efficiency
  - (i.e., miles per gallon);
- how is fuel efficiency effected by a car's weight?

To answer this question,

- you could obtain a dataset with several different car models, and
- attempt to identify a relationship between
  - weight (which we'll call wt) and
  - miles per gallon (which we'll call mpg).

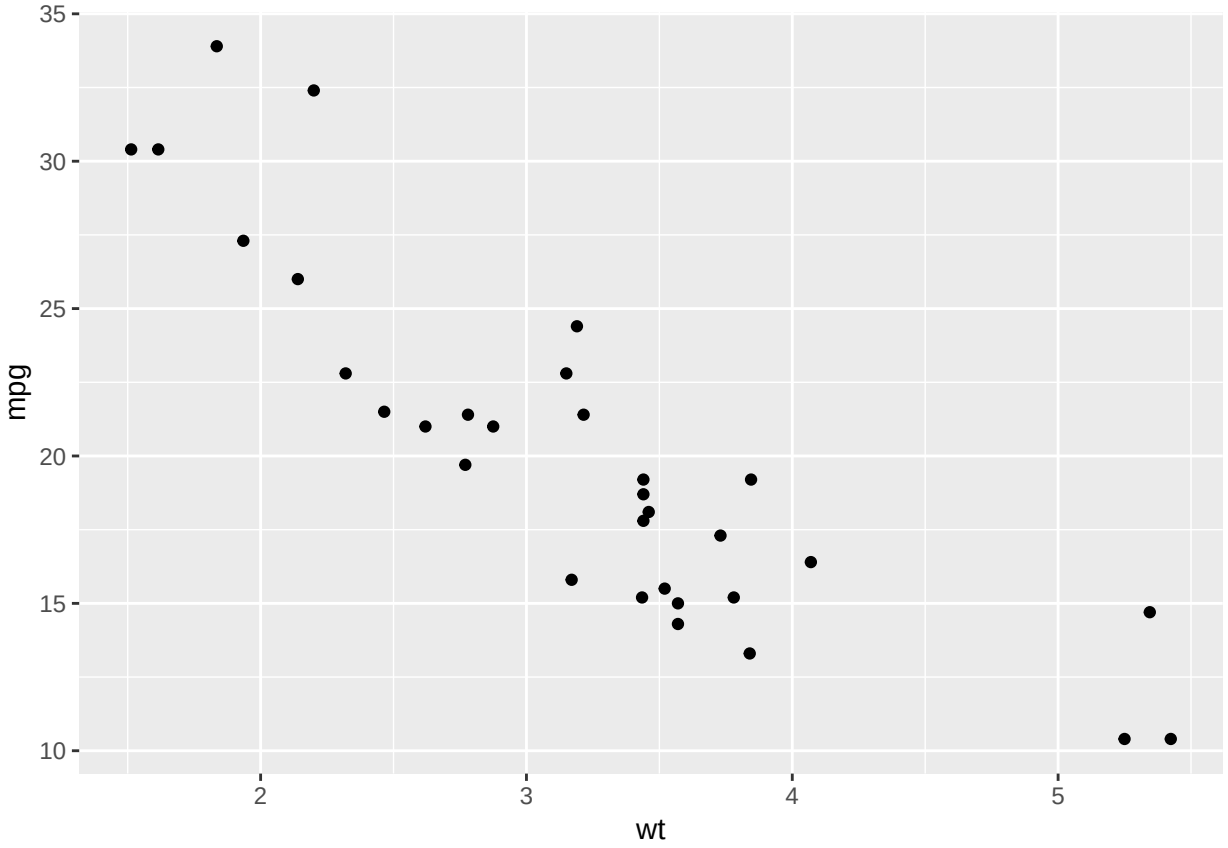
A good starting point would be some EDA

- simply plot the data,
- so first, we'll create a scatterplot using R's ggplot:

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
  geom_point()
```



Just examining the data visually,

- it's pretty clear that there's some relationship.

But if we want to make more precise claims

- about the relationship between wt and mpg,
- we need to specify this relationship mathematically.

So, as we press forward in our analysis,

- we'll make the assumption that
  - this relationship can be described mathematically;
- more precisely, we'll assume that
  - this relationship can be described by some mathematical function, $f(x)$.
- In the case of the above example, we'll be making the assumption
  - that "miles per gallon" can be described
  - as a function of "car weight".

Assuming this type of mathematical relationship,

- machine learning provides a set of methods
  - for identifying that relationship.

Said differently, machine learning provides a set of computational methods

- that accept data observations as inputs,
- and subsequently estimate that mathematical function, $f(x)$;
- machine learning methods learn the relationship
  - by being trained with an input dataset.

Ultimately, once we have this mathematical function (a model),

- we can use that model to make predictions and inferences.

14.2.1.8.3 How much math you really need to know

What we just discussed

- about "estimating functions" and "mathematical relationships"
  - might cause you to ask a question:
- "how much math do I need to know to do machine learning?"

Ok, here is some good news:

- to implement basic machine learning techniques,
- you don't need to know much math.

To be clear, there is quite a bit of math involved in machine learning,

- but most of that math is taken care of for you.

For the most part, R libraries and functions

- perform the mathematical calculations for you.

You just need to know

- which functions to use, and
- when to use them.

Here's an analogy: if you were a carpenter,

- you wouldn't need to build your own power tools.
  - your own drill and power saw.
- Therefore, you wouldn't need to understand
  - the mathematics, physics, and electrical engineering principles
  - that would be required to construct those tools from scratch.
- You could just go and buy them "off the shelf."
- To be clear, you'd still need to learn how to use those tools,
  - but you wouldn't need a deep understanding
  - of math and electrical engineering to operate them.

When you're first getting started with machine learning,

-the situation is very similar: - you can learn to use some of the tools, - without knowing the deep mathematics that makes those tools work.

Having said that, the above analogy is somewhat imperfect.

At some point, as you progress to more advanced topics,

- it will be very beneficial to know the underlying mathematics.

Ok, so you don't need to know that much math to get stared,

- but you're not entirely off the hook.
- you still need to know how to use the tools properly.

In some sense, this is one of the challenges

- of using machine learning tools in R:
- many of them are difficult to use.

R has many packages for implementing various machine learning methods,

- but unfortunately many of these tools were designed separately,
  - and they are not always consistent in how they work.

- The syntax for some of the machine learning tools is very awkward,
  - and syntax from one tool to the next is not always the same.
- If you don't know where to start, machine learning in R can become very confusing.

This is why the caret package is useful for machine learning in R.

14.2.1.8.4 A quick introduction to caret

For starters, let's discuss what caret is.

The caret package is a set of tools for building machine learning models in R.

The name "caret" stands for **Classification And REgression Training**.

As the name implies, the caret package gives you a toolkit

- for building classification models and regression models.

Moreover, caret provides you with essential tools for:

– Data preparation, including: - imputation, - centering/scaling data, - removing correlated predictors, - reducing skewness – Data splitting – Model evaluation – Variable selection

14.2.1.8.5 Caret simplifies machine learning in R

While caret has broad functionality,

- the real reason to use caret is that it's simple and easy to use.

As noted above, one of the major problems with machine learning in R

- is that most of R's different machine learning tools have different interfaces.
- They almost all "work" a little differently from one another:
  - the syntax is slightly different from one modeling tool to the next;
- Tools for different parts of the machine learning workflow
  - don't always "work well" together;
- tools for fine tuning models or performing critical functions
  - may be awkward or difficult to work with.
- Said succinctly, R has many machine learning tools,
  - but they can be extremely clumsy to work with.

Caret solves this problem.

To simplify the process,

- caret provides tools
  - for almost every part of the model building process,
- and moreover, provides a common interface
  - to these different machine learning methods.

For example, caret provides a simple, common interface

- to almost every machine learning algorithm in R.
- When using caret, different learning methods
  - like linear regression,
  - neural networks, and
  - support vector machines,
- all share a common syntax
  - (the syntax is basically identical, except for a few minor changes).

Moreover, additional parts of the machine learning workflow –

- like cross validation and parameter tuning – are built directly into this common interface.

To say that more simply,

- caret provides you with an easy-to-use toolkit for building many different model types and
  - executing critical parts of the ML workflow.
- This simple interface enables rapid, iterative modeling.

In turn, this iterative workflow will allow you to develop good models

- faster,
- with less effort, and
- with less frustration.

14.2.1.8.6 Caret's syntax

Now that you've been introduced to caret,

- let's return to the example above (of mpg vs wt)
- and see how caret works.

Again, imagine you want to learn the relationship between mpg and wt.

As noted above, in mathematical terms, this means

- identifying a function, $f(x)$,
  - that describes the relationship between wt and mpg.

Here in this example,

- we're going to make an additional assumption
  - that will simplify the process somewhat:
- we're going to assume that the relationship is linear;
- we'll assume that that it can be described
  - by a straight line of the form $f(x) = \beta_0 + \beta_1 x$.

In terms of our modeling effort,

- this means that we'll be using linear regression
  - to build our machine learning model.

Without going into the details of linear regression

- let's look at how we implement linear regression with caret.

14.2.1.8.7 The train() function

The core of caret's functionality is the train() function.

train() is the function that we use to "train" the model.

- That is, train is the function that
  - will "learn" the relationship between mpg and wt.

Let's take a look at this syntactically.

Here is the syntax for a linear regression model,

- regressing mpg on wt.

```
#~~~~~~~~~~~~~~~~~~~~~~~~
# Build model using train()
#~~~~~~~~~~~~~~~~~~~~~~~~
require(caret)
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
model.mtcars_lm <- train(mpg ~ wt
                         ,data = mtcars
                         ,method = "lm"
                         )
```

That's it. The syntax for building a linear regression

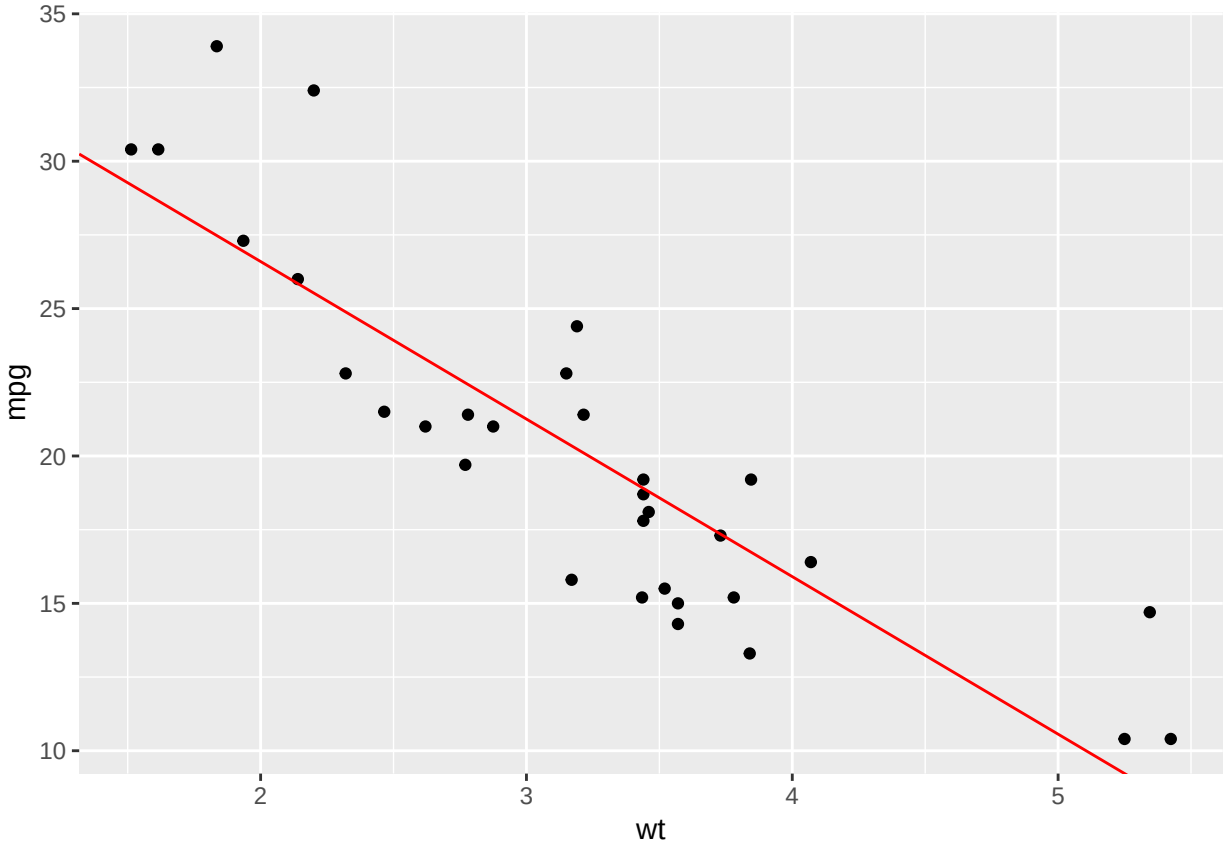
- is extremely simple with caret.

Now that we have a simple model,

- let's quickly extract the regression coefficients and
- plot the model
  - i.e., plot the linear function that describes
  - the relationship between mpg and wt

```
#~~~~~~~~~~~~~~~~~~~~~~~~~
# Retrieve coefficients for
#  - slope
#  - intercept
#~~~~~~~~~~~~~~~~~~~~~~~~~
coef.icept <- coef(model.mtcars_lm$finalModel)[1]
coef.slope <- coef(model.mtcars_lm$finalModel)[2]

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Plot scatterplot and regression line
#  using ggplot()
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_abline(slope = coef.slope, intercept = coef.icept, color = "red")
```



Now, let's look more closely at the syntax and how it works.

When training a model using train(), you only need to tell it a few things:

– The dataset you're working with – The target variable you're trying to predict (e.g., the mpg variable) – The input variable (e.g., the wt variable) – The machine learning method you want to use (in this case "linear regression")

14.2.1.8.8 Formula notation

In caret's syntax,

- you identify the target variable and
- input variables using the "formula notation."

The basic syntax for formula notation

- is y x,
  - where y is your target variable or response,
  - and x is your predictor.

Effectively, y x tells caret

- "I want to predict y
  - on the basis of a single input, x."

Now, with this knowledge about caret's formula syntax,

- let's reexamine the above code.
- Because we want to predict mpg on the basis of wt,
  - we use the formula mpg wt.

- Again, this line of code is the "formula"
  - that tells train() our target response variable and our input predictor variable.
- If we translate this line of code into English,
  - we're effectively telling train(),
  - "build a model that predicts mpg (miles per gallon) on the basis of wt (car weight)."

14.2.1.8.9 The data = parameter

The train() function also has a data = parameter.

This basically tells the train() function

- what dataset we're using to build the model.

Said differently,

- if we're using the formula mpg ~ wt
  - to indicate the target and predictor variables,
- then we're using the data = parameter
  - to tell caret where to find those variables.

So basically, data = mtcars

- tells the caret function that the data and]
- the relevant variables
- can be found in the mtcars dataset.

14.2.1.8.10 The method = parameter

Finally, we see the method = parameter.

- This parameter indicates what machine learning method
  - we want to use to predict y.
- In this case, we're building a linear regression model, - so we are using the argument "lm".

Keep in mind, however, we could select a different learning method.

- Although it's beyond the current scope
  - to discuss all of the possible learning methods that we could use here,
  - there are many different methods we could use.
- For example, if we wanted to use the k-nearest neighbor technique,
  - we could use the "knn" argument instead.
- If we did that, train() would still predict mpg on the basis of wt,
  - but would use a different statistical technique to make that prediction.
- This would yield a different model; a model that makes different predictions.

As you learn more about machine learning, and

- want to try out more advanced machine learning techniques,
  - this is how you can implement them.
- You simply change the learning method
  - by changing the argument of the method = parameter.

This is a good place to reiterate one of caret's primary advantages:

- switching between model types
  - is extremely easy when we use caret's train() function.
- Again, if you want to use linear regression to model your data,
  - you just type in "lm" for the argument to method =;
- if you want to change the learning method to k-nearest neighbor,

  - you just replace “lm” with “knn”.

Caret’s syntax allows you to very easily change the learning method.

- In turn, this allows you to “try out” and evaluate
  - many different learning methods rapidly and iteratively.
- You can just re-run your code with different values for the method parameter,
  - and compare the results for each method.

14.2.1.8.11 Next steps

Now that you have a high-level understanding of caret,

- you’re ready to dive deeper into machine learning in R.

Keep in mind though,

- if you’re new to machine learning,
- there’s still lots more to learn.

Machine learning is intricate and fascinatingly complex.

Moreover, caret has a variety of additional tools for model building.

We’ve just scratched the surface here.

14.2.1.9 The Perceptron (Neural Networks)

Lets start looking into the broad topic of Neural Networks for Machine Learning

Artificial Neural Networks (ANN)

- are the supervised learning techniques
- whose logic is similar to biological neural systems.

A simple ANN technique is the single-layer perceptron and

- it is a classification technique
- estimating a binary attribute
- whose value can be 0 or 1.

The single layer perceptron models

- are as shown in the following figure:

The perceptron works like a neuron

- in the sense that it sums the impact
- of all the inputs and outputs to 1
- if the sum is above a defined threshold.

The model is based on the following parameters:

- A weight for each feature, defining its impact
- A threshold above which the estimated output is 1

Starting from the features, the model estimates the attribute through these steps

- Compute the output through a linear regression:
  - multiply each feature by its weight and sum all of them
- Estimate the attribute
  - to 1 if the output is above the threshold
  - and to 0 otherwise

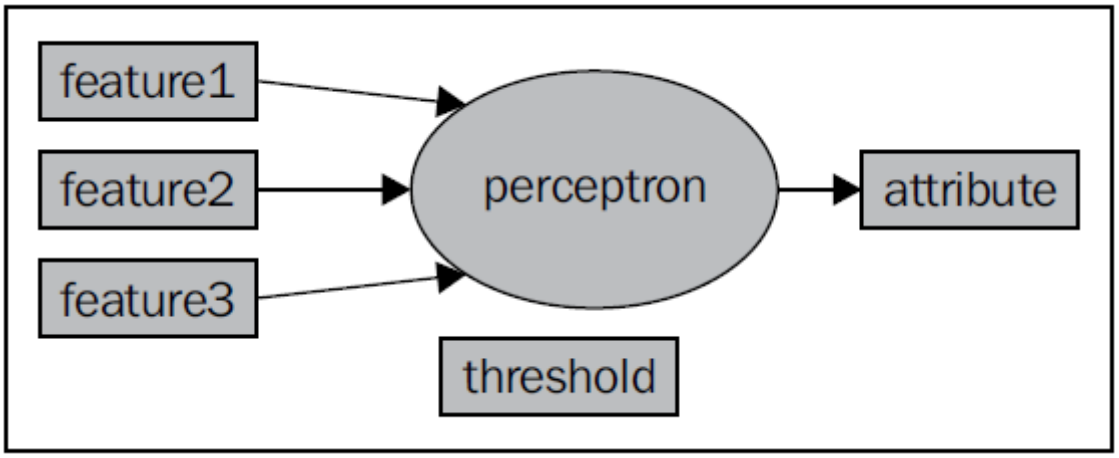


Figure 2: RMLE-7p10.png

14.2.1.10 Links for Information in This Document

14.2.1.10.1 R Machine Learning Essentials, by Michele Usuelli, Packt Publishing, 2014

14.2.1.10.2 A quick introduction to machine learning in R with caret