



Statistical Learning in
Marketing
EBM214A05.2022-2023.1

FREE EDITION*

SUMMARY OF EVERYTHING FROM WEEK 1

LECTURES, CODE EXAMPLES, TIPS & STEP-BY-STEP ASSIGNMENT 1 GUIDE.

Note: This course has no readings.

Enhanced with a dynamic table of contents.

For the full version 

rug mads madlad



100%

OF THE PROFIT FROM THIS
SUMMARY IS DONATED TO
THE FOLLOWING NGOs:



This summary helps you study & people in need.

Please do not share it for free, but ask your
friends to buy it and do a good deed.

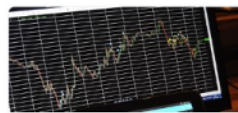
MADS MADLAD

Note from MADS MADLAD:

Thank you for checking out my free summary. When I was writing these I sometimes struggled with this program, but there were no summaries available. This is why I decided to write something that is truly complete with a lot of effort put into it.

It helped me and my friends get good grades, but I always had you in mind, the future reader. When necessary, I always went the extra mile to make my summaries, more readable, organized and complete.

If you feel like it, leave me a review of how the course is going using this summary, it will make my day to hear your feedback.

Check out my other extensive summaries for other MADS courses:

Statistical Learning in
Marketing
EBM214A05.2022-2023.1



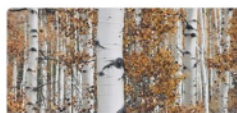
Data Engineering for
MADS
EBM213A05.2022-2023.1



Companies, Brands, and
Consumers
EBM215A05.2022-2023.1



Retail & Omnichannel
Marketing
EBM880B05.2022-2023.1



Market Models
EBM077A05.2022-2023.1



Data Science Methods for
MADS
EBM216A05.2022-2023.1



Digital Marketing
Intelligence
EBM079B05.2022-2023.1

Contact info:

If you need help or have an inquiry, contact me: <https://www.georgedreemer.com>

Connect with me on LinkedIn: <https://www.linkedin.com/in/georgedreemer/>

Donations:

By no means am I looking for fellow students to send me money! But if you feel like sending me some ETH or BTC, you can do so here:

--> **ETH:** 0x123e086c6808459e7fC6Ac7F64a77dBA1dDe0149

--> **BTC:** bc1qgwzc82vph5v8rmzef4ywechjf85772n7m2e22g

MADS MADLAD

wishes you good luck & perseverance.



Grades Testimony:

COURSE CODE	TITLE	SCORE	DATE	RESULT
EBS001A10	Business Research Methods for Pre-MSc	8	21-12-2021	8
EBS002A05	Mathematics for Pre-MSc	9	10-11-2021	9
EBS003A05	Organization Theory & Design for Pre-MSc	7	05-11-2021	7
EBB098A05	Contemporary Theories on Business and Management	6	11-05-2022	6
EBB649C05	Strategic Management B&M	8	15-06-2022	8
EBB617B05	Human Resource Management B&M	8	08-04-2022	8
EBB104A05	Behavioural Decision Making	7	03-11-2021	7
EBB085A05	Marketing Research for E&BE	8	04-04-2022	8
EBS008B10	Research Paper for Pre-MSc Marketing	7	05-07-2022	7
EBM043A05	Business Ethics	8	14-11-2022	8
EBB105B05	Digital Marketing Analytics	8	21-01-2022	8
EBM213A05	Data Engineering for MADS	7	01-11-2022	7
EBM214A05	Statistical Learning in Marketing	8	02-11-2022	8
EBM215A05	Companies, Brands, and Consumers	8	05-11-2022	8
EBM216A05	Data Science Methods for MADS	9	20-01-2023	9

Table of Contents

Week 1 (Lecture 1).....	10
Lecture 1 – Intro to R + Intermediate R.....	10
Introduction to R: Variables, Vectors, Matrices, Factors, Data Frames and Lists.....	10
Variables	10
Vectors.....	10
Matrices	10
Factors	11
Data Frames (DF).....	11
Lists	12
Intermediate R: Conditionals, Control flow, Loops, Functions, Apply family, Utilities.	12
Conditionals and Control Flow	12
- Relational operators	12
- Logical operators	13
- Conditional Statements	13
Loops.....	14
- While loops	14
- For loops	14
Functions.....	15
Apply family	16
- Lapply	16
- Sapply.....	16
- Vapply	17
Utilities.....	17
- Data utilities	18
- Importing Data in R	19
Week 2 (Lecture 2).....	20
Lecture 2 – Reducing data complexity	20

Concepts in this lecture:	20
Principal Component Analysis.....	20
Sequence of action in PCA.....	20
Terminology	23
In Practice – How to in R:	23
Exploratory Factor Analysis (EFA)	25
Sequence of action in EFA	25
Estimation process	26
In Practice – How to in R:	28
Multidimensional scaling (MDS)	32
In Practice – How to in R:	33
Final Remarks:	34
Week 3 (Lecture 3).....	35
Lecture 3 – General linear model, part A: Introduction/ANOVA/ANCOVA	35
Assumptions to be satisfied for OLS:	37
Goodness of fit (R^2)	37
Interpreting Adjusted- R^2	37
Other Goodness of Fit Indices	38
Statistical Inference.....	38
Possible reasons for insignificance	39
In Practice (the case) – How to in R:	40
ANOVA.....	45
Full vs. Reduced model	47
In Practice – How to in R (continued):	47
ANCOVA.....	48
Homogeneity of regression slopes	49
In practice – How to in R (continued):	51
Week 4 (Lecture 4).....	52
Lecture 4 – General linear model, part B: Multiple Regression	52
Multiple Linear Regression	52

Interpretation: variable transformations.....	52
Judging the impact of variables.....	53
Standardized coefficients.....	54
Multicollinearity.....	54
Indications of multicollinearity: How to test for it?.....	55
Variance Inflation Factors & Tolerance.....	55
Solutions for Multicollinearity:.....	56
Dummy Variable Trap.....	56
Moderation Effects.....	58
Types of Moderation Effects.....	58
Types of Moderator Variables.....	58
Interpreting Moderation Effects.....	59
Rescaling of Variables.....	59
Is the presence of multicollinearity always a problem?.....	60
Good Practices.....	60
In Practice – How to in R:.....	61
Week 5 (Lecture 5 and 6).....	66
Lecture 5 – Modern Time Series Analysis, part A.....	66
2 Types of Time Series.....	66
6 Methodological Steps.....	67
In Practice – How to in R (pre- 6 Methodological Steps above):.....	67
1. Granger causality.....	69
How many lags to include?.....	69
Back to Practice – How to in R (1. Granger Causality):.....	70
2. Unit Root and Cointegration Tests.....	71
Autoregressive process.....	71
Role of Φ (phy).....	72
Stationarity.....	73
Condition for Stationarity:.....	73

Cointegration	74
Testing for Cointegration.....	75
Back to Practice – How to in R:.....	76
Cointegration in Marketing	80
4 long-term scenarios.....	80
VAR (Vector Autoregressive) Model	81
Lecture 6 – Modern Time Series Analysis, part B	82
VAR.....	82
3 Forms of VAR models	82
Structural VAR (SVAR)	82
Reduced-form VAR.....	83
In Practice – How to in R	83
Endogenous vs. Exogenous (1).....	84
Types of Exogeneity	84
Lag Selection (2).....	85
Lag Selection (continued)	87
Symmetry vs. Asymmetries	87
Dealing with (non-)stationarity	87
Vector Error Correction (VEC).....	88
Seasonality in VAR/VEC	88
IRF.....	90
Parameter numerosity	90
Back to Practice – How to in R:.....	91
Back to Practice – How to in R (IRF):.....	93
FEVD: Forecast Error Variance Decomposition	97
Week 6 (Lecture 7).....	99
Lecture 7 – Cluster Analysis	99
Cluster Analysis: Introduction	99
Steps to Cluster Analysis	100
Step 1: Define the Research Purpose	100

Step 2: Selecting a distance measure	102
In Practice – How to in R	103
Step 3: Doing the Clustering.....	104
Hierarchical Clustering: Agglomerative vs. Divisive	104
Agglomerative Methods: linkage method	105
Agglomerative Methods: centroid method	105
Agglomerative Methods: Ward’s method	106
Back to Practice – How to in R.....	106
Mean-based Clustering: k-means method.....	112
Back to Practice – How to in R.....	113
Hierarchical vs. Mean-based Clustering	114
Hierarchical Clustering	114
Mean-based Clustering	114
Model-based Clustering.....	116
In Practice – How to in R	116
Testing Cluster Solutions: Interpreting and Labeling the Clusters	117
Testing the Differences: Are observed differences significant? - ANOVA. 118	
Where are the differences? - TukeyHSD.....	119
In Practice – How to in R (Tukey).....	120
Week 7 (Lecture 8).....	122
Lecture 8 – Doing Good Research	122
Independence.....	122
Honesty	122
Gathering data:	122
Treating data:.....	123
Analyzing data:.....	123
Reporting results:.....	123
Validity.....	124
Types of Validity.....	124
Types of Research and their Validity	124

Reliability & Reproducibility.....	125
Correlation vs. Causality	125
Rigor vs. Relevance	127
EXTENSIVE STEP-BY-STEP GUIDE FOR TEAM ASSIGNMENT 1	129

Factors: specific types of variables that is categorical in nature.

- Values are like labels (e.g. Rural, Urban...)
- Only a limited number of categories

```
# Temperature
temperature_vector <- c("High", "Low", "High", "Low", "Medium")
factor_temperature_vector <- factor(temperature_vector, order = TRUE, levels = c(
  "Low", "Medium", "High"))
factor_temperature_vector
```

Data Frames (DF): multi-dimensional array with different data types.

When inspecting a df there are some handy basic functions:

- head() – prints the header and first few rows of your data frame
- tail() – prints the header and last few rows of the data frame
- str() – tells you about the structure of you df, namely:
 - total number of observations
 - total number of variables
 - full list of the variable names
 - data type of each variable
 - the first observartion/row

```
script.R
3   "Mars", "Jupiter", "Saturn",
4   "Uranus", "Neptune")
5   type <- c("Terrestrial planet",
6           "Terrestrial planet",
7           "Terrestrial planet",
8           "Terrestrial planet", "Gas giant",
9           "Gas giant", "Gas giant", "Gas giant")
10  diameter <- c(0.382, 0.949, 1, 0.532,
11              11.209, 9.449, 4.007, 3.883)
12  rotation <- c(58.64, -243.02, 1, 1.03,
13              0.41, 0.43, -0.72, 0.67)
14  rings <- c(FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, TRUE)
15
16  # Create a data frame from the vectors
17  planets_df <- data.frame(name, type, diameter, rotation, rings)
```

Lists: a list of items with different length, type, characteristics.

You can put: vectors, matrices and even data frames in a list!

```
script.R
1 # Vector with numerics from 1 up to 10
2 my_vector <- 1:10
3
4 # Matrix with numerics from 1 up to 9
5 my_matrix <- matrix(1:9, ncol = 3)
6
7 # First 10 elements of the built-in data frame mtcars
8 my_df <- mtcars[1:10,]
9
10 # Adapt list() call to give the components names
11 my_list <- list(my_vector, my_matrix, my_df)
12 names(my_list) <- c("vec", "mat", "df")
```

Intermediate R: Conditionals, Control flow, Loops, Functions, Apply family, Utilities.

Conditionals and Control Flow

- *Relational operators:* comparing R objects
 - Equality ==
 - Greater/Less Than > / <
 - Not Equal !=

```
script.R
1 # Comparison of logicals
2 TRUE == FALSE
3
4 # Comparison of numerics
5 -6*14 != 17 - 101
6
7 # Comparison of character strings
8 "user" == "user"
9
10 # Compare a logical with a numeric
11 TRUE == 1
```

- *Logical operators:*

- AND &
- OR |

script.R

```
1 # The linkedin and last variable are already defined for you
2 linkedin <- c(16, 9, 13, 5, 2, 17, 14)
3 last <- tail(linkedin, 1)
4
5 # Is last under 5 or above 10?
6 last < 5 | last > 10
7
8 # Is last between 15 (exclusive) and 20 (inclusive)?
9 last > 15 & last <= 20
```

- *Conditional Statements:*

- if
- else if
- else

script.R

```
1 # Variables related to your last day of recordings
2 li <- 15
3 fb <- 9
4
5 # Code the control-flow construct
6 if (li >= 15 & fb >= 15) {
7   sms <- 2 * (li + fb)
8 } else if (li < 10 & fb < 10) {
9   sms <- 0.5 * (li + fb)
10 } else {
11   sms <- li + fb
12 }
13
14 # Print the resulting sms to the console
15 print(sms)
```

Loops

- *While loops*: look like repeated if statements

```

script.R
1  # Initialize the speed variable
2  speed <- 64
3
4  # Extend/adapt the while loop
5  while (speed > 30) {
6    print(paste("Your speed is", speed))
7    if (speed > 48 ) {
8      print("Slow down big time!")
9      speed <- speed - 11
10   } else {
11     print("Slow down!")
12     speed <- speed - 6
13   }
14 }

```

```

R Console    Slides
[1] "Your speed is 64"
[1] "Slow down big time!"
[1] "Your speed is 53"
[1] "Slow down big time!"
[1] "Your speed is 42"
[1] "Slow down!"
[1] "Your speed is 36"
[1] "Slow down!"

```

- *For loops*: iterate over all elements in a sequence.
 - o **Two ways**: normal and indexed:

```

script.R
1  # The linkedin vector has already been defined for you
2  linkedin <- c(16, 9, 13, 5, 2, 17, 14)
3
4  # Loop version 1
5  for ( p in linkedin) {
6    print(p)
7  }
8
9
10
11 # Loop version 2
12 for (i in 1:length(linkedin)) {
13   print(linkedin[i])
14 }

```

Functions: a defined operation that does something with the input you give it.

R offers both in-built functions and the ability to define our own functions.

- Pre-defined functions example: **mean()**

```
script.R
1 # The linkedin and facebook vectors have already been created for you
2 linkedin <- c(16, 9, 13, 5, 2, 17, 14)
3 facebook <- c(17, 7, 5, 16, 8, 13, 14)
4
5 # Calculate the mean of the sum
6 avg_sum <- mean(linkedin + facebook)
7
8 # Calculate the trimmed mean of the sum
9 avg_sum_trimmed <- mean(linkedin + facebook, trim = 0.2)
10
11 # Inspect both new variables
12 print(avg_sum)
13 print(avg_sum_trimmed)
```

- Self-made function examples:

```
script.R
3 # Define the interpret function
4 interpret <- function(num_views) {
5   if (num_views > 15) {
6     print(paste("You're popular!"))
7     return(num_views)
8   }
9   } else {
10    print(paste("Try to be more visible!"))
11    return(0)
12  }
13 }
14
```

```
10 # Create a function sum_abs()
11 sum_abs <- function(a,b){
12   abs(a)+abs(b)
13 }
```

Apply family:

- *Lapply*: apply a function to a specified set of data, and return a list.

```

script.R
1 # The vector pioneers has already been created for you
2 pioneers <- c("GAUSS:1777", "BAYES:1702", "PASCAL:1623", "PEARSON:1857")
3
4 # Split names from birth year
5 split_math <- strsplit(pioneers, split = ":")
6
7 # Convert to lowercase strings: split_low
8 split_low <- lapply(split_math, tolower)
9
10 # Take a look at the structure of split_low
11 str(split_low)

```

Run Code

```

R Console Slides
str(split_low)

List of 4
 $ : chr [1:2] "gauss" "1777"
 $ : chr [1:2] "bayes" "1702"
 $ : chr [1:2] "pascal" "1623"
 $ : chr [1:2] "pearson" "1857"

```

- *Sapply*: apply a function to a specified set of data, and return a vector.

```

12 # Use sapply() to find each day's maximum temperature
13 sapply(temp, max)

```

```

R Console Slides
[1] 9

# Use sapply() to find each day's maximum temperature
sapply(temp, max)

[1] 9 13 8 7 9 9 9

```


- *Vapply*: more robust version of *Sapply*, where you have more control over the output.

```

script.R
1 # temp is already available in the workspace
2
3 # Definition of basics()
4 basics <- function(x) {
5   c(min = min(x), mean = mean(x), max = max(x))
6 }
7
8 # Apply basics() over temp using vapply()
9 vapply(temp, basics, numeric(3))

```

```

R Console Slides
# Apply basics() over temp using vapply()

vapply(temp, basics, numeric(3))

      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
min  -1.0   5 -3.0 -2.0  2.0 -3.0  1.0
mean  4.8   9  2.2  2.4  5.4  4.6  4.6
max   9.0  13  8.0  7.0  9.0  9.0  9.0

```

Utilities

R contains a large number of useful utilities and functions to do basic work on your data. You can also add external libraries with specialized utilities.

- **Mathematical utilities:**
 - `abs()` : Calculate the absolute value.
 - `sum()` : Calculate the sum of all the values in a data structure.
 - `mean()` : Calculate the arithmetic mean.
 - `round()` : Round the values to 0 decimal places by default. Try out `?round` in the console for variations of `round()` and ways to change the number of digits to round to.

- *Data utilities:*

- `seq()` : Generate sequences, by specifying the `from` , `to` , and `by` arguments.
- `rep()` : Replicate elements of vectors and lists.
- `sort()` : Sort a vector in ascending order. Works on numerics, but also on character strings and logicals.
- `rev()` : Reverse the elements in a data structures for which reversal is defined.
- `str()` : Display the structure of any R object.
- `append()` : Merge vectors or lists.
- `is.*()` : Check for the class of an R object.
- `as.*()` : Convert an R object from one class to another.
- `unlist()` : Flatten (possibly embedded) lists to produce a vector.
 - **Grep and grepl:** R's Regex - check whether a regular expression could be matched with a character vector.

```

script.R  solution.R  Light Mode
1 # The emails vector has already been defined for you
2 emails <- c("john.doe@ivyleague.edu", "education@world.gov", "dalai.lama@peace.org",
3             "invalid.edu", "quant@bigdatacollege.edu", "cookie.monster@sesame.tv")
4
5 # Use grepl() to match for .edu addresses more robustly
6 grepl("@.*\\.edu$", emails)
7
8 # Use grep() to match for .edu addresses more robustly, save result to hits
9 hits <- grep("@.*\\.edu$", emails)
10
11 # Subset emails using hits
12 emails[hits]
  
```

```

R Console  Slides
hits <- grep("@.*\\.edu$", emails)

# Subset emails using hits

emails[hits]

[1] "john.doe@ivyleague.edu"  "quant@bigdatacollege.edu"
  
```

- **Sub and gsub:** takes it one step further from grep/grepl, by both matching and then replacing the match with what you assigned.

```
script.R Light Mode
1 # The emails vector has already been defined for you
2 emails <- c("john.doe@ivyleague.edu", "education@world.gov", "global@peace.org",
3            "invalid.edu", "quant@bigdatacollege.edu", "cookie.monster@sesame.tv")
4
5 # Use sub() to convert the email domains to datacamp.edu
6 sub("@.*\\.edu$", "@datacamp.edu", emails)
```

↻ Run Code Submit Answer

```
R Console Slides
# Use sub() to convert the email domains to datacamp.edu
sub("@.*\\.edu$", "@datacamp.edu", emails)

[1] "john.doe@datacamp.edu"      "education@world.gov"
[3] "global@peace.org"          "invalid.edu"
[5] "quant@datacamp.edu"         "cookie.monster@sesame.tv"
```

- *Importing Data in R /w Readr package*
 - **Read_csv**
 - **Read_tsv**
 - **Read_delim**

Note from MADS Madlad: When you have done the Datacamp courses Intro to R and Intermediate R yourself, you will get a better feel for all of this than just by reading it. It's also way more fun!

Good luck and congrats on finishing your first week of the MADS program.