


Week 6: *Conjoint Questions*

 EMSE 6035: Marketing Analytics for Design Decisions

 John Paul Helveston

 October 05, 2022

Some RMarkdown tips

Convert a data frame to a markdown table with `kable()`

```
library(tidyverse)
```

```
mtcars %>%  
  kable()
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

Example from last year

```
library(tidyverse)
library(here)

df <- read_csv(here("data", "competitors.csv"))
df %>%
  kable()
```

brand	volumelnOz	quantity	price	pricePerCup	biodegradability	opacity	logo
greatValue	9	50	\$2.98	\$0.06	FALSE	TRUE	FALSE
decorRack	9	50	\$5.99	\$0.12	FALSE	TRUE	FALSE
tigerChef	9	100	\$7.99	\$0.08	FALSE	TRUE	FALSE
smartly	9	80	\$2.79	\$0.03	FALSE	FALSE	FALSE
solo	9	50	\$4.04	\$0.08	FALSE	FALSE	FALSE
greatValue	9	100	\$3.76	\$0.04	FALSE	FALSE	FALSE
ecoProducts	9	1000	\$187.69	\$0.19	TRUE	FALSE	TRUE
worldCentric	9	2000	\$220.00	\$0.11	TRUE	FALSE	TRUE
hefty	18	50	\$3.98	\$0.08	FALSE	TRUE	FALSE

More `kable()` formatting options:
`{kableExtra}` package

References

Simple approach: Insert a footnote with `^[]`

markdown

```
The Eiffel Tower is 324 meters tall^[From the [Eiffel Tower  
wikipedia page](https://en.wikipedia.org/wiki/Eiffel_Tower)]
```

render

The Eiffel Tower is 324 meters tall¹

¹From the [Eiffel Tower wikipedia page](https://en.wikipedia.org/wiki/Eiffel_Tower)

References

Complex (but more complete) approach: Use bibtex

<https://bookdown.org/yihui/rmarkdown-cookbook/bibliography.html>

You can insert citations with `[@citekey]`, and a "References" table will be automatically created.

Footnotes are perfectly fine for this class

Week 6: *Conjoint Questions*

1. Defining choice questions in R
2. Displaying choice questions in RMarkdown

QUIZ 2

3. Choice questions in formr

Week 6: *Conjoint Questions*

1. Defining choice questions in R
2. Displaying choice questions in RMarkdown

QUIZ 2

3. Choice questions in formr

Download the `logitr-cars` repo from GitHub

emse-madd-gwu / `logitr-cars` Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags

Go to file Add file Code

Clone

HTTPS SSH GitHub CLI

`https://github.com/emse-madd-gwu/logitr-cars`

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

jhelvy finished mxl models		
code	finished mxl models	
data	finished mxl models	
figs	updated code to use {maddTools}	
models	finished mxl models	
sims	updated code to use {maddTools}	
.gitignore	added simulated data	3 months ago
README.md	Update README.md	last month
logitr-cars.Rproj	update Rproj file name	last month

We'll be using the `{cbcTools}` package today

```
install.packages("remotes")  
remotes::install_github("jhelvy/cbcTools")
```

Choice question components

1. Generate **profiles** for each attribute and level
2. Create a survey **design** data frame from **profiles**

Basic survey

Any combination of attributes can be shown in each choice question

Question 1

Option:	1	2	3
Price:	\$25	\$20	\$25
Fuel Economy:	30 (mpg)	20 (mpg)	30 (mpg)
Accel. Time:	8 (s)	7 (s)	7 (s)
Powertrain:	Electric	Electric	Electric

Question 2

Option:	1	2	3
Price:	\$15	\$25	\$25
Fuel Economy:	20 (mpg)	20 (mpg)	25 (mpg)
Accel. Time:	6 (s)	7 (s)	8 (s)
Powertrain:	Electric	Gasoline	Electric

Labeled design

One attribute is used as the "label" - choice options are fixed according to the label

Question 1

Option:	Gasoline	Electric
Price:	\$20	\$15
Fuel Economy:	30 (mpg)	25 (mpg)
Accel. Time:	6 (s)	6 (s)

Question 2

Option:	Gasoline	Electric
Price:	\$20	\$20
Fuel Economy:	30 (mpg)	25 (mpg)
Accel. Time:	7 (s)	7 (s)

Design with a "None" option

A "none" option means they can choose an "other" option

Question 1

Option:	1	2	3	None
Price:	\$25	\$25	\$20	
Fuel Economy:	20 (mpg)	25 (mpg)	25 (mpg)	
Accel. Time:	8 (s)	6 (s)	8 (s)	
Powertrain:	Electric	Electric	Gasoline	

Question 2

Option:	1	2	3	None
Price:	\$25	\$20	\$20	
Fuel Economy:	30 (mpg)	20 (mpg)	25 (mpg)	
Accel. Time:	6 (s)	7 (s)	8 (s)	
Powertrain:	Electric	Electric	Electric	

Open `logitr-cars.Rproj`

Attribute-specific features

Some attributes may only be valid for certain levels of other attributes

Example: The driving range of an electric vehicle (EV) only applies to EVs and not gasoline-powered vehicles.

To implement this, edit `profiles` prior to using `cbc_design()`

Your Turn

20:00

1. With your team, discuss the specific choice question design for your project
 - Regular or labeled?
 - Include a "none" option (outside good) or not?
1. Edit the `make_choice_questions.R` file to design your choice questions.

Week 6: *Conjoint Questions*

1. Defining choice questions in R
2. **Displaying choice questions in RMarkdown**

QUIZ 2

3. Choice questions in formr

Displaying your choice questions online

1. Export your choice questions as a .csv file
2. Upload your .csv file somewhere (e.g. GitHub)
3. Use R code to extract the values to display
4. Use RMarkdown to display the values

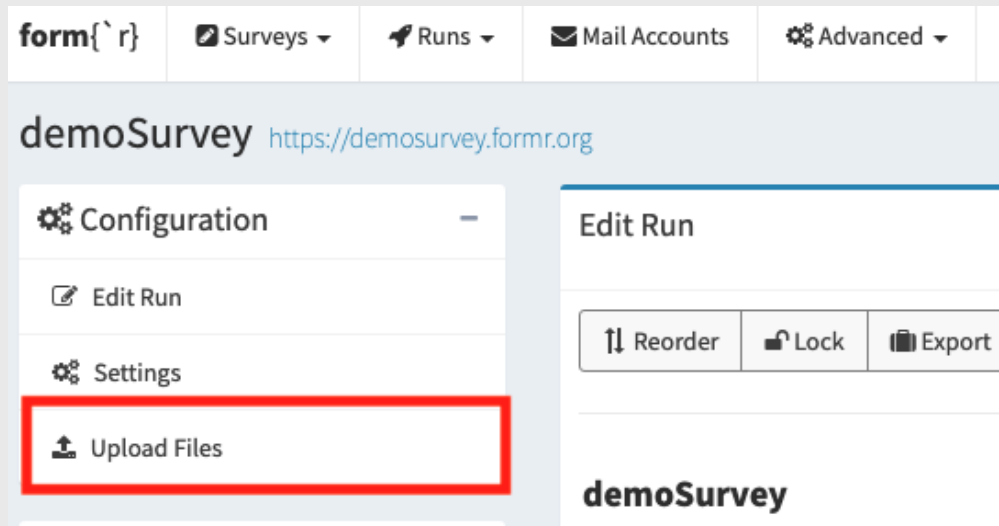
1. Export your choice questions as a .csv file

```
write_csv(design, here('choice_questions.csv'))
```

2. Upload your .csv file somewhere

Inside a formr run (private)

github.com (public)



apples example

Use R code to extract the values to display

- Read `choice_questions.csv` from web
- Randomly choose a respondent ID
- Filter rows for that respondent ID
- Serialize the data frame to json format

Side note on serializing a data frame

Converts a data frame to one long string

```
df
```

```
#>   altID profileID price fuelEconomy accelTime powertrain
#> 1     1         54    25          30         8    Electric
#> 2     2         38    20          20         7    Electric
#> 3     3         45    25          30         7    Electric
```

```
df_json <- jsonlite::toJSON(df)
df_json
```

```
#>
[{"altID":1,"profileID":54,"price":25,"fuelEconomy":30,"accelTime":8,"powertrain":"Electric"}
{"altID":2,"profileID":38,"price":20,"fuelEconomy":20,"accelTime":7,"powertrain":"Electric"}
{"altID":3,"profileID":45,"price":25,"fuelEconomy":30,"accelTime":7,"powertrain":"Electric"}
```

Use RMarkdown to display the values

Create separate data frames for each alternative

```
library(dplyr)
alts <- jsonlite::fromJSON(df_json)
alt1 <- alts %>% filter(altID == 1)
alt2 <- alts %>% filter(altID == 2)
alt3 <- alts %>% filter(altID == 3)
```

Use RMarkdown formatting to display content in each alternative

```
**Option 1**
**Price**: $ `r alt1$price`
**Powertrain**: $ `r alt1$powertrain`
**Fuel Economy**: `r alt1$fuelEconomy` mpg
**0-60 Accel. Time**: `r alt1$accelTime` s
```

Option 1

Price: \$ 25

Powertrain: \$ Electric

Fuel Economy: 30 mpg

0-60 Accel. Time: 8 s

Show options in a table with `kable()`

```
library(dplyr)

alts <- jsonlite::fromJSON(df_json) %>%
  # Add $ sign to price
  mutate(price = scales::dollar(price))
%>%
  # Make nicer attribute labels
  select(
    `Option:`           = altID,
    `Powertrain:`       = powertrain,
    `Price:`             = price,
    `Fuel Economy (mpg):` = fuelEconomy,
    `Accel. Time (s):`   = accelTime)

# Drop row names
row.names(alts) <- NULL
```

Display the *transpose*, `t(alts)`

```
kable(t(alts))
```

Option:	1	2	3
Powertrain:	Electric	Electric	Electric
Price:	\$25	\$20	\$25
Fuel Economy (mpg):	30	20	30
Accel. Time (s):	8	7	7

Download the `formr4conjoint` repo from GitHub

(code used in the related [blog post](#))

`jhelvy / formr4conjoint` Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

jhelvy added package installs to readme

figs	added package installs to readme
survey	added consent form content in p1
.gitignore	Update .gitignore
LICENSE.md	Create LICENSE.md
README.Rmd	added package installs to readme
README.md	added package installs to readme 20 minutes ago
formr4conjoint.Rproj	Init 2 years ago

Clone ?

HTTPS SSH GitHub CLI

`https://github.com/jhelvy/formr4conjoi`

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

Your Turn

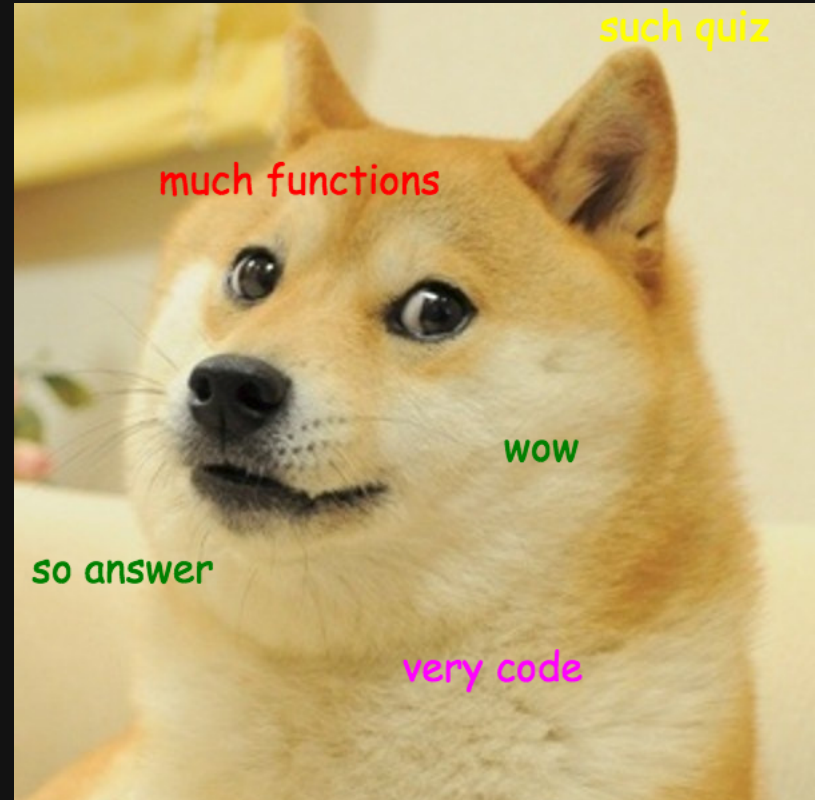
1. With your team, upload your `choice_questions.csv` file somewhere online (e.g. inside a formr run or on a GitHub repo).
2. Edit the `p2-choice-questions.Rmd` or `p2-choice-questions-table.Rmd` file to implement your choice questions in RMarkdown.

You should be able to knit the file to visually test how one of your choice questions is rendering.

Quiz 2

Link is in the #class channel

10:00



Week 6: *Conjoint Questions*

1. Defining choice questions in R
2. Displaying choice questions in RMarkdown

QUIZ 2

3. Choice questions in formr

Your first few rows

- Read `choice_questions.csv` from web
- Randomly choose a respondent ID
- Filter rows for that respondent ID
- Serialize the data frame to json format

Using the `calculate` type (example sheet)

RMarkdown

```
# Read in the choice questions
library(tidyverse)
design <-
  read_csv("https://raw.githubusercontent.com/jhelvy/formr4conjoint/master/survey/questions.csv")

# Define the respondent ID
respondentID <- sample(design$respID, 1)

# Create the subset of rows for that respondent ID
df <- design %>%
  filter(respID == respondentID) %>%
  mutate(image =
    paste0("https://raw.githubusercontent.com/jhelvy/formr4conjoint/master/survey/images/",
           image))

# Convert df to json
df_json <- jsonlite::toJSON(df)
```

Google sheet

C	D	E	K
type	optional	name	value
calculate		time3	Sys.time()
calculate		survey	library(tidyverse) read_csv("https://raw.githubusercontent.com/jhelvy/formr4conjoint/master/survey/questions.csv")
calculate		respondentID	sample(survey\$respID, 1)
calculate		df	survey %>% filter(respID == respondentID) %>% mutate(image = paste0("https://raw.githubusercontent.com/jhelvy/formr4conjoint/master/survey/images/", image))
calculate		df_json	jsonlite::toJSON(df)

Random choice questions as **buttons** (example sheet)

Use the `mc_button` question type




label

- Show your question text
- Insert a code chunk to create one-row data frame for each alternative

choice columns

- Insert RMarkdown code to display each alternative

(1 of 8) If these were your only options, which would you choose?

Option 1	Option 2	Option 3
		
Type: Gala Price: \$ 3.5 / lb Freshness: Excellent	Type: Fuji Price: \$ 4 / lb Freshness: Poor	Type: Pink Lady Price: \$ 3.5 / lb Freshness: Poor

Random choice questions as **table** (example sheet)




- Use the `mc_button` question type

label

- Show your question text
- Insert a code chunk to modify `alts` data frame & display it using `kable()`
- Use `kableExtra` to control table styling

choice columns

- Simple text / number for each option

Option:	1	2	3
			
Price:	\$4.00 / lb	\$1.50 / lb	\$1.00 / lb
Type:	Fuji	Gala	Gala
Freshness:	Average	Average	Poor

Option 1	Option 2	Option 3
----------	----------	----------

Your Turn

1. Discuss the layout you would prefer to implement for your choice questions (buttons or table).
2. Make a Google Sheet using your team Google account to start implementing your conjoint questions.

[buttons example sheet](#)

[table example sheet](#)