Final Presentation

Collin Brehmer, Molly Hischke, Kyle Hancock, Nikki Johnson

12/13/2019

Research Question

What positions on the Jeopardy board are most likely to be a Daily Double by round and across years?

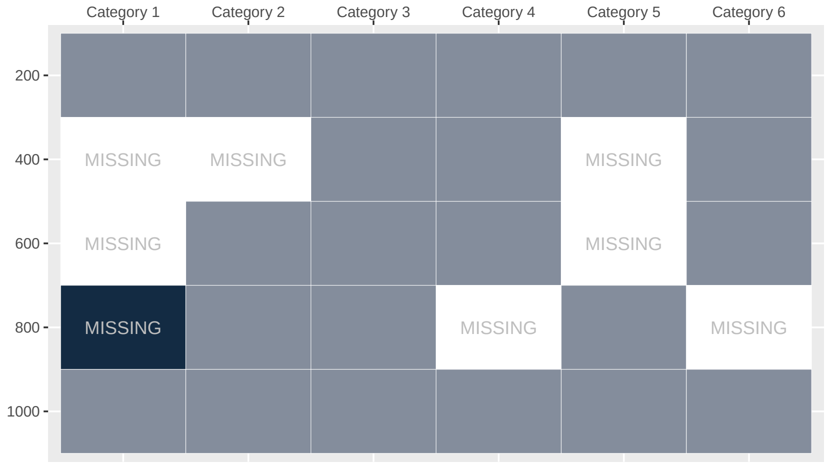
Introduction

- Importance to the game
    - Arthur Chu (2014)
    - James Holzhauer (2019)
- Previous analyses
    - None across years
- Challenge of finding position
- Visual representation of results

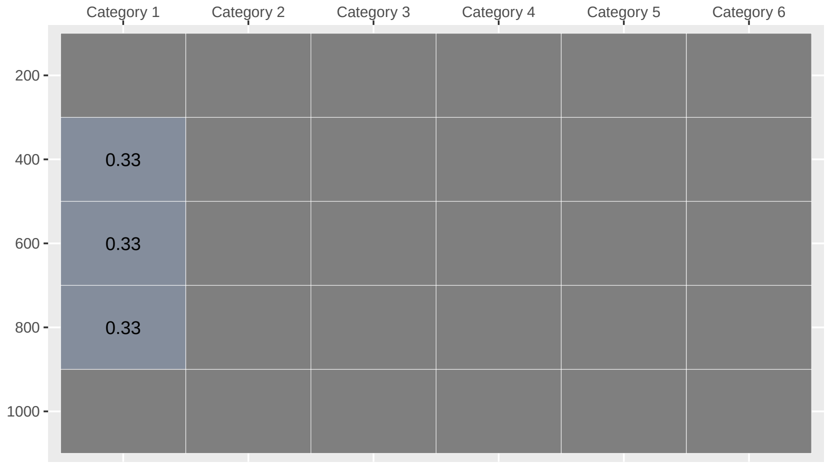
Methods

- Only included episodes from dates beyond November 26, 2001
- Finding x position of daily double (filtered to episodes with 12 categories)
- Finding y position of daily double (Weighted across missing values)

Methods


Category 1
Category 2
Category 3
Category 4
Category 5
Category 6
200
400
600
800
1000
MISSING
MISSING
MISSING
MISSING
MISSING
MISSING
MISSING
MISSING

Methods


Category 1
Category 2
Category 3
Category 4
Category 5
Category 6
200
400
600
800
1000
0.33
0.33
0.33

Results

Flex Dashboard:

- Daily Double Round 1 since 2001
- Daily Double Round 2 since 2001
- Daily Double Location by year
- Daily Doubles are not evenly distributed across the board
- No differences in daily double location between Round 1 and Round 2
- 15 out of 19 years had the most daily doubles within the 800/1600 row

Conclusions

- Similar to previous analyses
  - https://flowingdata.com/2015/03/03/where-to-find-jeopardy-daily-doubles/
- Shift in location in 2016
  - Arthur Chu and subsequent analyses
- Future locations (pattern or random)

Overview of Approach

Nest data by air date

Pull out subsets of that data

Modify subsets by:

Adding additional information (i.e. weight, x/y position)

Manipulating data (i.e. expanding, further subsetting)

Rejoin dataframes to get desired information

Create tidy dataset for plots

Interesting Packages or Techniques

- Creating perfect data frame
  - Package: `tidyr`
  - Function(s): `expand()`
- Mapping
  - Package: `purrr`
  - Function(s): `map()`, `map2()`
- Joining
  - Package: `dplyr`
  - Function(s): `right_join()`, `full_join()`, `left_join()`
- Plotting
  - Package: `plotly`
  - Function(s): `plot_ly()`

Examples

- Input
  - categories_asked: information about each episode

```
mutate(categories_unique = map(data, ~ unique(select(., cat
       categories_unique = map(categories_unique,
                               ~ rownames_to_column(., var = '
        perfect_pos = map2(categories_unique, categories_a
                               ~ full_join(.x, .y)),
        perfect_pos = map(perfect_pos,
                              ~ group_by(., round)),
        perfect_pos = map(perfect_pos,
                              ~ expand(., x_pos, value)))
```

- Output
  - categories_unique: x position assigned to each category
  - perfect_pos: perfect df for each game

Examples

- Input
  - dd_perfect: perfect DD df
  - dd_weight: df containing the weight to be assiged to DD questions

```
mutate(dd_perfect = map2(dd_perfect, dd_weight,
                  ~ full_join(.x, .y, by = "category")),
       dd_perfect = map(dd_perfect,
                   ~ mutate(., daily_double = case_when(
                     is.na(daily_double) == TRUE ~ weight,
                     is.na(daily_double) == FALSE ~ daily_dou
```

- Output
  - dd_perfect: perfect DD df with weights added

Examples

```r
daily_double_year %>%
  unnest() %>%
  plot_ly(
    x = ~ x_pos,
    y = ~ y_pos,
    z = ~ Percent,
    frame = ~ year,
    hovertemplate = paste("Daily Double Percent: %{z:,}%<br
                          "<extra></extra>"),
    colors = "Blues",
    type = 'heatmap'
) %>%
  layout(title = list(text = ""),
         xaxis = list(title = "",
                      side = 'top'),
          yaxis = list(title = ""))
```

Lessons Learned

1. Seemingly simple problems may not be simple to answer.
2. Pros/cons of `ggplot()` vs `plot_ly()`
3. Project organization and GitHub
4. Writing other code before data is completely clean is beneficial.