Washingtonian Top 100 Restaurant Picker

Sungjoo Cho and Kate Lamoreaux

2023-12-13

GitHub Repository: https://github.com/katelmrx/cho-lamoreaux-a1.git

R Shiny app: DCRestaurantPicker folder - app.R file

Introduction

Question of Interest: How can we optimize restaurant choices in the Washington area?

Our research aims to explore the Washingtonian Magazine's "Top 100 Restaurants" list, by analyzing factors including restaurant rankings, location, cuisine, and pricing. We propose to develop an interactive graphic, which will allow users to explore the data and recommend restaurants based on their preferences. Our goal is to provide a straightforward guide for both locals and tourists, making it easier for them to discover and enjoy dining experiences in the greater Washington D.C. area. Table 1 shows the first six rows of the dataset obtained from Washingtonian Web Scraping data.

Data

Dataset

To achieve our research goal, we will utilize a dataset comprising essential information about restaurants. This dataset includes details such as restaurant names, rankings, genres, phone numbers, pricing, ratings, websites, locations, as well as latitude and longitude. To compile this comprehensive dataset, we plan to obtain information from three multiple sources: Washingtonian, Yelp, and Google Places.

Methodology

1. Washingtonian Web Scraping

We initially acquired data by web scraping the Washingtonian Magazine website, extracting details on the top 100 ranked restaurants in the area, using R packages rvest and httr. This information involves restaurant names, ranking, genres, addresses, and phone numbers. Table 1 below shows the first six columns of dataset obtained from Washingtonian Web Scraping.

```
## # A tibble: 6 x 5
##   ranking name                   genre            address        ‘phone number‘
##   <chr>   <chr>                  <chr>            <chr>          <chr>
## 1 1.      "The Dabney "          "Mid-Atlantic "  "122 Blagden ~ (202) 450-1015
## 2 2.      "Albi "                "Levantine "     "1346 Fourth ~ <NA>
## 3 3.      "L’Ardente "           "Italian "       "200 Massachu~ <NA>
## 4 4.      "Gemini "               "Mediterranean " "1509 17th St~ <NA>
## 5 5.      "Pineapple and Pearls " "American "      "715 Eighth S~ (202) 595-7375
## 6 6.      "Inferno Pizzeria "     "Pizza "         "12207 Darnes~ (301) 963-0115
```

2. Yelp API Data

Subsequently, we obtained ratings and price level information using the Yelp API with Python and libraries such as certify, chrset, and fuzzywuzzy. The associated script, titled script.py, is available in the Data folder on GitHub.

```
## # A tibble: 6 x 8
##   name                categories  rating price address   phone latitude longitude
##   <chr>               <chr>        <dbl> <chr> <chr>     <dbl>    <dbl>     <dbl>
## 1 The Dabney          New Americ~    4.5 "$$$~ 122 Bl~ 1.20e10     38.9     -77.0
## 2 The Dabney Cellar   Bars, New ~    4.5 ""    1222 9~ 1.20e10     38.9     -77.0
## 3 Albi                Lebanese       4.5 ""    1346 4~ 1.20e10     38.9     -77.0
## 4 L'Ardente           Italian        4.5 ""    200 Ma~ 1.20e10     38.9     -77.0
## 5 Pineapple & Pearls New Americ~    4.5 "$$$~ 715 8t~ 1.20e10     38.9     -77.0
## 6 Sushi Nakazawa      Japanese, ~    4.5 "$$$~ 1100 P~ 1.20e10     38.9     -77.0
```

3. Merging Washingtonian Web Scraping Data and Yelp API Data

The Washingtonian web scraping data and Yelp API data were then merged in R using the left_join function from the dplyr package. During this process, discrepancies were found in restaurant names between Washingtonian and Yelp data, such as 'Pineapple and Pearls' in Washingtonian versus 'Pineapple & Pearls' in Yelp. To address this, a new variable, 'key_name', was created to standardize names by removing special characters and spaces for accurate matching using the gsub function. The table below presents the initial six rows of the merged dataframe, combining data from both the Washingtonian and Yelp datasets.

```
## # A tibble: 6 x 13
##   ranking name        genre genre_washingtonion categories_yelp address    phone
##   <chr>   <chr>       <chr> <chr>               <chr>           <chr>      <dbl>
## 1 1.      The Dabney  Amer~ Mid-Atlantic        New American, ~ 122 Bl~  1.20e10
## 2 2.      Albi        Leba~ Levantine           Lebanese        1346 F~  1.20e10
## 3 3.      L'Ardente   Ital~ Italian             Italian         200 Ma~  1.20e10
## 4 4.      Gemini      Medi~ Mediterranean       <NA>            1509 1~ NA
## 5 5.      Pineapple ~ Amer~ American            New American    715 Ei~  1.20e10
## 6 6.      Inferno Pi~ Ital~ Pizza               <NA>            12207 ~  1.30e10
## # i 6 more variables: rating_yelp <dbl>, rating_google <dbl>, price_yelp <chr>,
## #   price_google <chr>, latitude <dbl>, longitude <dbl>
```

4. Google Places API Data

Finally, we obtained restaurant details including latitude, longitude, Google rating, Google price levels, user rating totals, and website information using the Google Places API with the R googleway package. Particularly, the **'google_places'** function was used to collect latitude, longitude, and address data. Additionally, **'google_places_details'** function was employed to get information on rating, price level, website, user rating total, and phone number.

```
## # A tibble: 6 x 9
##   name        address latitude longitude rating price_level phone_number website
##   <chr>       <chr>      <dbl>     <dbl>  <dbl>       <dbl> <chr>        <chr>
## 1 The Dabney  122 Bl~     38.9     -77.0    4.7           3 (202) 240-2~ http:/~
## 2 Albi        1346 4~     38.9     -77.0    4.8           1 <NA>         http:/~
## 3 L'Ardente   200 Ma~     38.9     -77.0    4.6           3 (202) 448-0~ https:~
## 4 Gemini      1509 1~     38.9     -77.0    4.9          NA <NA>         http:/~
## 5 Pineapple ~ 715 8t~     38.9     -77.0    4.8           2 (202) 595-7~ https:~
## 6 Inferno Pi~ 12207 ~     39.1     -77.3    4.5           2 (301) 963-0~ http:/~
## # i 1 more variable: user_rating_total <dbl>
```

5. Merge Google API data to Washingtonian & Yelp combined dataset

This dataset was merged with the combined dataset from Washingtonian and Yelp, by using the left_join function based on the name of the restaurant. Then we created new variables, rating_avg and price_avg, which indicate the average rating and price level derived from both Yelp and Google Places data. Moreover, we identified missing data in the phone numbers and website information. Thus, we manually imputed the missing information by conducting lookups when applicable. This integrated dataset was used for our analysis, where we aim to unveil key factors contributing to the popularity of these restaurants and to develop an interactive graphic. With this dataset, we developed a function designed to take user inputs regarding cuisine preferences, budget considerations, and ratings. This function filters and presents restaurant options from the dataset that align with the specified criteria.

```
## # A tibble: 6 x 18
##      ID top25_wm ranking name   genre phone_number price_yelp price_yelp_cleaned
##   <dbl>    <dbl> <chr>   <chr>  <chr> <chr>        <chr>                   <dbl>
## 1     1        1 1.      The D~ Amer~ (202) 240-2~ $$$$                        4
## 2     2        1 2.      Albi   Leba~ (202) 921-9~ <NA>                       NA
## 3     3        1 3.      L'Ard~ Ital~ (202) 448-0~ <NA>                       NA
## 4     4        1 4.      Gemini Medi~ <NA>         <NA>                       NA
## 5     5        1 5.      Pinea~ Amer~ (202) 595-7~ $$$$                        4
## 6     6        1 6.      Infer~ Ital~ (301) 963-0~ <NA>                       NA
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

Characteristics of the Dataset

The final dataset consists of 100 rows of individual restaurants ranked by Washingtonian Magazine. The variables include ID, top25_wm, ranking, names, genre, phone_number, price_yelp, rating_yelp, price_google, rating_google, price_avg, rating_avg, address, latitude, longitude, google_user_rating_total, and website.

Table 1: Variable Description

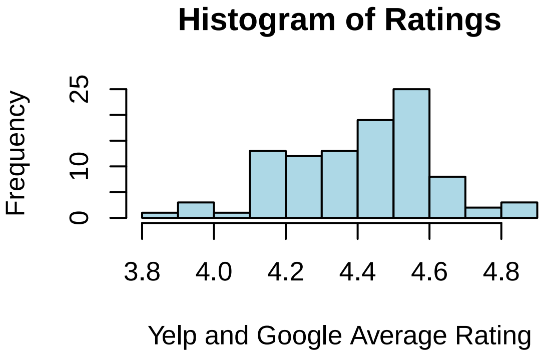
Variable Name	Variable Description	Value
ID	Restaurant ID number	1-100
top25_wm	Indication of whether the restaurant belongs to the top 25 restaurants in the Washington DC area	1 for yes; 0 otherwise
ranking	Ranking within the top 25 restaurants in Washington DC	1-25, or NA if not in the top 25
names	The name of the restaurants	chr
genre	The type of the restaurants	chr
phone_number	Contact numbers for restaurants	chr
price_yelp	Price level on Yelp	$-$$$$

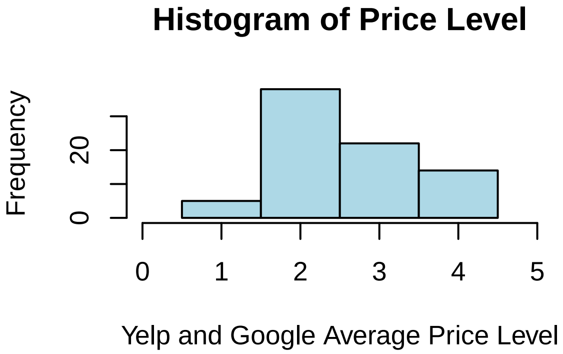
Variable Name	Variable Description	Value
price_yelp_cleaned	Numeric price level on Yelp	1(less expensive) – 4(more expensive)
rating_yelp	The rating on Yelp	1(poor) – 5(good)
price_google	Price level on Google	1(less expensive) – 4(more expensive)
rating_google	The rating on Google	1(poor) – 5(good)
price_avg	The average price level across Yelp and Google	1(less expensive) – 4(more expensive)
rating_avg	The average rating across Yelp and Google	1(poor) – 5(good)
address	Location details of the restaurant	chr
latitude	Latitude of the restaurant's address	num
longitude	Longitude of the restaurant's address	num
website	The website link of the restaurant	chr
google_user_rating_total	The number of reviews for the restaurant on Google	num

Analysis

Basic Data Analysis

The histogram of ratings indicates that the majority of top 100 restaurants in Washington DC tend to have ratings from 4.4 to 4.6. Additionally, the histogram of price level shows that the most prevalent price level among these restaurants is between 2 and 3.





The table below shows that the Halal restaurant has the highest rating of 4.7 among the top 100 restaurants in the area, accompanied by a price level of 1. Fine dining follows closely with a mean rating of 4.667 and an average price of 4. The American restaurant category comprises the largest share in the top 100 restaurants in Washington DC, featuring 14 restaurants, followed by Italian with 13 and Japanese with 10 restaurants.

Table 2: Rating and Price by Genre

genre	mean_rating	mean_price	genre_count
Halal	4.700	1.000	1
Fine dining	4.667	4.000	3
Afghan	4.600	2.000	2
European	4.600	4.000	1
Laotian	4.600	2.000	1
Lebanese	4.600	2.250	2
Spanish/Japanese	4.600	3.000	1
South American	4.575	3.000	2

genre	mean_rating	mean_price	genre_count
American/European	4.550	2.500	1
American/Italian	4.550	3.000	1
Bar	4.550	2.000	3
Korean	4.550	2.000	2
French/American	4.525	3.750	2
French	4.517	3.167	6
Chinese/Korean	4.500	2.000	1
Ethiopian	4.500	2.000	1
Iranian	4.500	2.500	1
Italian/American	4.500	2.000	1
Mexican	4.500	2.000	1
American	4.454	3.083	14
Barbecue	4.450	1.500	2
Thai	4.450	2.000	3
Chinese	4.420	2.250	5
Yemeni	4.400	2.000	1
Italian	4.377	2.962	13
Caribbean	4.350	2.500	2
Indian	4.333	2.167	3
Spanish	4.320	3.000	5
Seafood	4.300	2.000	2
Japanese	4.285	2.643	10
Cambodian/Taiwanese	4.250	2.000	1
Vietnamese	4.200	1.000	2
Malaysian	4.100	2.000	1

Function to optimize restaurant choices

We developed a function to optimize restaurant choices according to user input. The **‘find_restaurants’** function takes four arguments - genre, rating, price_level, and top24_wm. The function then effectively filters and returns restaurant options from the dataset that align with the user-defined criteria. The following is an example of how the function can be used.

```r
find_restaurants <- function(genre = NULL, rating = NULL, price_level = NULL, top25_wm = NULL) {

  # use a copy of the original dataset
  result <- restaurants

  # filter based on genre
  if (!is.null(genre)) {
    result <- result[result$genre == genre, ]
  }

  # filter based on rating
  if (!is.null(rating)) {
    result <- result[result$rating_avg >= rating, ]
  }

  # filter based on price_level
  if (!is.null(price_level)) {
    result <- result[result$price_avg == price_level, ]
  }

  # filter based on top25_wm
  if (!is.null(top25_wm)) {
    result <- result[result$top25_wm == top25_wm, ]
  }

  result <- result %>% filter(!is.na(name))
  return(result)
}
```

```
# example usage:
find_restaurants(genre="Japanese", rating=4, price_level=3, top25_wm=0)
```

```
## # A tibble: 1 x 18
##      ID top25_wm ranking name   genre phone_number price_yelp price_yelp_cleaned
##   <dbl>    <dbl> <chr>   <chr>  <chr> <chr>        <chr>                     <dbl>
## 1    73        0 <NA>    Nasime Japa~ (703) 548-1~ <NA>                         NA
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

```
find_restaurants(genre="Korean", rating=4, top25_wm=1)
```

```
## # A tibble: 1 x 18
##      ID top25_wm ranking name  genre  phone_number price_yelp price_yelp_cleaned
##   <dbl>    <dbl> <chr>   <chr> <chr>  <chr>        <chr>                     <dbl>
## 1    13        1 13.     Anju  Korean (202) 845-8~ <NA>                         NA
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

```
find_restaurants(genre="Italian", top25_wm=1)
```

```
## # A tibble: 5 x 18
##      ID top25_wm ranking name   genre phone_number price_yelp price_yelp_cleaned
##   <dbl>    <dbl> <chr>   <chr>  <chr> <chr>        <chr>                     <dbl>
## 1     3        1 3.      L’Ard~ Ital~ (202) 448-0~ <NA>                         NA
## 2     6        1 6.      Infer~ Ital~ (301) 963-0~ <NA>                         NA
## 3    16        1 16.     2 Amys Ital~ (202) 885-5~ <NA>                         NA
## 4    17        1 17.     Carus~ Ital~ (202) 661-0~ <NA>                         NA
## 5    21        1 21.     Fiola~ Ital~ (202) 350-4~ $$$$                          4
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

```
find_restaurants(genre="Mexican")
```

```
## # A tibble: 1 x 18
##      ID top25_wm ranking name   genre phone_number price_yelp price_yelp_cleaned
##   <dbl>    <dbl> <chr>   <chr>  <chr> <chr>        <chr>                   <dbl>
## 1    30        0 <NA>    Anafre Mexi~ (202) 758-2~ $$                          2
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

```r
find_restaurants(top25_wm=1)
```

```
## # A tibble: 25 x 18
##       ID top25_wm ranking name  genre phone_number price_yelp price_yelp_cleaned
##    <dbl>    <dbl> <chr>   <chr> <chr> <chr>        <chr>                   <dbl>
##  1     1        1 1.      The ~ Amer~ (202) 240-2~ $$$$                        4
##  2     2        1 2.      Albi  Leba~ (202) 921-9~ <NA>                       NA
##  3     3        1 3.      L'Ar~ Ital~ (202) 448-0~ <NA>                       NA
##  4     4        1 4.      Gemi~ Medi~ <NA>         <NA>                       NA
##  5     5        1 5.      Pine~ Amer~ (202) 595-7~ $$$$                        4
##  6     6        1 6.      Infe~ Ital~ (301) 963-0~ <NA>                       NA
##  7     7        1 7.      Sush~ Japa~ (202) 462-8~ <NA>                       NA
##  8     8        1 8.      Causa Sout~ (202) 629-3~ <NA>                       NA
##  9     9        1 9.      Sush~ Japa~ (202) 289-3~ $$$$                        4
## 10    10        1 10.     Xiqu~ Span~ (202) 913-4~ <NA>                       NA
## # i 15 more rows
## # i 10 more variables: rating_yelp <dbl>, address <chr>, latitude <dbl>,
## #   longitude <dbl>, rating_google <dbl>, price_google <dbl>, rating_avg <dbl>,
## #   price_avg <dbl>, website <chr>, google_user_rating_total <dbl>
```

Interactive Graphic

We developed an interactive graphic using the R Shiny app framework, and the code is contained in the 'app.R' file.

Conclusion

Based on the analysis above,