

Data Technologies Group 1 Update 11/18/2021

- Tyler Antoloci
- Ragan Astle
- Nate Nellis
- McKade Thomas

Image Processing

- All ads have a grid like layout that facilitates cropping
- May attempt to filter all non-black colors to help remove product images
- Front Page advertises dollars off, while rest of the ad advertises % off.

VOL. 6 NO. 2 JANUARY 20 - FEBRUARY 2

 <p>Oscar Mayer® 16 oz. Beef Cotto Salami or Bologna SAVE 26%</p>	 <p>Oscar Mayer® 3.2 oz. Turkey or Ham Lunchables® SAVE 21%</p>	 <p>Velveeta® Original Cheese Slices 16 ct. SAVE 40%</p>	 <p>Progresso® 18.5 - 19 oz. Traditional or Light Chicken Noodle Soup SAVE 41%</p>
 <p>Progresso® 18.5 - 19 oz. Select Varieties Low Sodium Chicken Soup SAVE 41%</p>	 <p>Progresso® Traditional Chicken & Herb Dumplings Soup 18.5 oz. SAVE 41%</p>	 <p>Nabisco® 16 oz. Original or Unsalted Tops Saltine Crackers SAVE 18%</p>	 <p>Nabisco® 8 oz. Basil & Garlic or Rosemary & Jalapeño Triscuit® Crackers SAVE 23%</p>
 <p>Nabisco® 7 oz. Original or Pepper & Olive Oil Organic Triscuit® Crackers SAVE 23%</p>	 <p>Nabisco® 3.5 - 3.75 oz. Salt & Pepper or Sweet Potato Good Thins® Crackers SAVE 30%</p>	 <p>Nabisco® 3.5 oz. Three Cheese Good Thins® Crackers SAVE 23%</p>	 <p>Keebler® 9.5 oz. Parmesan Cheese & Basil or Sea Salt Pita Crackers SAVE 34%</p>
 <p>Keebler® 11.7 - 13.7 oz. Original or Reduced Fat Club® Crackers SAVE 34%</p>	 <p>Super Pretzel® Soft Baked Pretzels 6 ct. SAVE 23%</p>	 <p>Nabisco® Ritz® Bacon Crisp & Thins™ Chips 7.1 oz. SAVE 31%</p>	 <p>Nabisco® Ritz® Salt & Vinegar Crisp & Thins™ Chips 7.1 oz. SAVE 31%</p>

your COMMISSARY

DISCLAIMER: PICTURES ARE FOR ILLUSTRATION ONLY. THE PRODUCTS INCLUDED IN THIS SALE MAY BE DIFFERENT FROM THOSE PICTURED. SOME ITEMS MAY NOT BE AVAILABLE AT ALL LOCATIONS. EFFECTIVE FOR CONUS STORES ONLY.

Page 16

```

1 library(tesseract)
2 library(magick)
3 library(magrittr)
4 library(imagefx)
5 library(pdftools)
6 library(magick)
7
8 ad_test <- image_read("C:/Users/n8nel/Downloads/2752687-1200-100000.jpg")
9 text_ad_test <- ocr(ad_test)[]
10 cat(text_ad_test)
11
12 a1 <- image_crop(ad_test, geometry = geometry_area(width = 300,
13                                                    height = 150,
14                                                    x_off = 50,
15                                                    y_off = 590))
16 a2 <- image_crop(ad_test, geometry = geometry_area(width = 300,
17                                                    height = 125,
18                                                    x_off = 50,
19                                                    y_off = 975))
20 a3 <- image_crop(ad_test, geometry = geometry_area(width = 300,
21                                                    height = 125,
22                                                    x_off = 50,
23                                                    y_off = 1325))
24 b1 <- image_crop(ad_test, geometry = geometry_area(width = 275,
25                                                    height = 150,
26                                                    x_off = 330,
27                                                    y_off = 590))
28 b2 <- image_crop(ad_test, geometry = geometry_area(width = 300,
29                                                    height = 125,
30                                                    x_off = 330,
31                                                    y_off = 960))
32 b3 <- image_crop(ad_test, geometry = geometry_area(width = 300,
33                                                    height = 130,
34                                                    x_off = 330,
35                                                    y_off = 1310))
36 c1 <- image_crop(ad_test, geometry = geometry_area(width = 275,
37                                                    height = 150,
38                                                    x_off = 600,
39                                                    y_off = 575))
40 c2 <- image_crop(ad_test, geometry = geometry_area(width = 275,
41                                                    height = 125,
42                                                    x_off = 620,
43                                                    y_off = 975))
44 c3 <- image_crop(ad_test, geometry = geometry_area(width = 300,
45                                                    height = 150,
46                                                    x_off = 620,
47                                                    y_off = 1300))
48 d1 <- image_crop(ad_test, geometry = geometry_area(width = 275,

```

```

90 image_write(ad_test, "2752687-1200-100000.pdf", format = "pdf")
91
92 # based on https://www.r-bloggers.com/2017/08/tesseract-and-magick-high-q
93
94 ad_textProcessed <- ad_test %>%
95   image_resize("2000") %>%
96   image_convert(colorspace = "gray") %>%
97   image_trim()
98
99 ad_textOCR <- ad_textProcessed %>%
100   image_ocr()
101
102 cat(ad_textOCR)
103
104
105 ad_textOCRData <- ad_textProcessed %>%
106   image_ocr_data()
107
108 print(ad_textOCRData, n = 10)
109
110
111 ### - this one is the best
112
113 library(pdftools)
114
115 ad_textOCR2 <- pdf_ocr_text("2752687-1200-100000.pdf")
116
117 cat(ad_textOCR2)
118

```

Image Processing Code

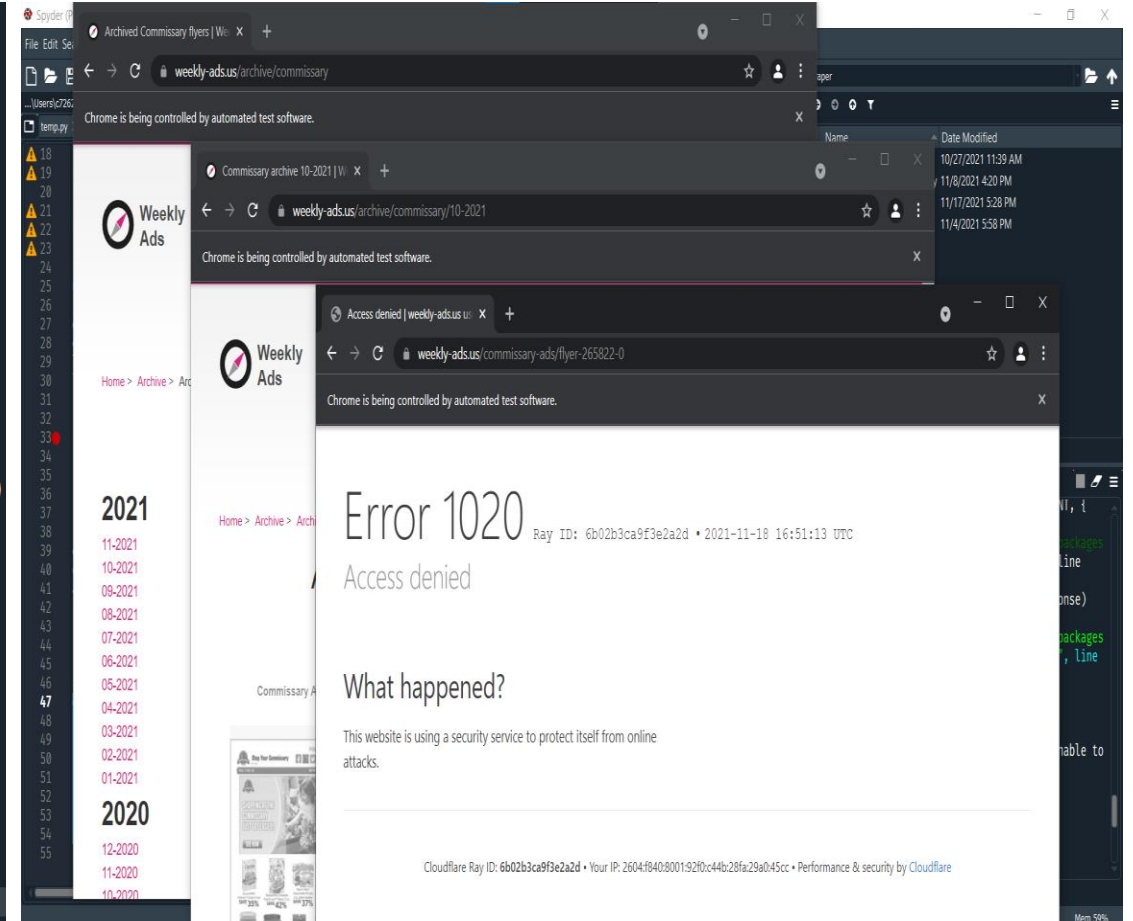
Data Collection

- Weekly-ads.us continues to thwart our efforts at scraping.
- Have successfully downloaded single jpeg with script, but implementing loops results in ip blocking.
- Using multiple web browsers helped but didn't get us all the way.
- Will be attempting to automate tor to randomize ip address throughout scraping process.

```

26 url = "https://weekly-ads.us/archive/commissary"
27
28 driver = webdriver.Chrome() # link to web driver connected to chromium
29 driver.get(url) # access site
30 links = driver.find_elements_by_tag_name('a')
31 link_urls = [] # initialize list of urls to target
32 regex = "^\\d{2}-\\d{4}$" # regex pattern for mm-yyyy date format
33
34 for link in links: # check each link in list of all links
35     if bool(search(regex, link.text)): # If link follows date format
36         link_urls.append(link.get_attribute('href')) # Then add url to url list
37
38
39
40 driver2 = webdriver.Chrome() # initializing multiple drivers prevents IP blocking
41 driver2.get(link_urls[1]) # follow first link that follows date format
42 ad_links = driver2.find_elements_by_xpath("//div[@class = 'leaflet-detail']/a[starts-with(@href, '/commissary-ads/flyer-')][1]")
43
44 for link in ad_links:
45     link.get_attribute('href')
46
47 # Single Ad Multiple Pages for each page
48 driver3 = webdriver.Chrome()
49 image_urls = [] # initialize list of url's leading to jpegs to be downloaded in bash
50
51 for link in ad_links:
52     driver3.implicitly_wait(0.5) #rapid requests trigger IP blocking, wait .5 seconds
53     driver3.get(link.get_attribute('href')) # retrieve url to jpeg
54     driver3.implicitly_wait(0.5)
55     image_urls.append(driver3.find_element_by_id("leaflet").get_attribute('src'))# ad to jpeg's url to list
56

```



Data Collection Code