دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

بازیابی پیشرفته اطلاعات

امیرمهدی کوششی

توضیح پروژه

در این تمرین من سعی کردم با برداشتن اطلاعاتی از یک سایت ایرانی، روی آن یک سری پیش پردازش‌های متنی انجام دهم و سپس به آنالیز کردن آن متون بپردازم. من کامنت‌هایی که افراد برای رستوران‌های فروشگاه آنلاین غذایی اسنپ‌فود گذاشته بودند را برداشتم، و بین سپس روی آن‌ها یک سری آمارگیری‌ها انجام دادم که در ادامه شرح خواهم داد.

پیدا کردن داده

من ابتدا با استفاده از ویژگی inspect element نرم‌افزار گوگل کروم، api هایی که با توجه به آن ها میشد کامنت‌های هر رستوران را در آورد را پیدا کرده، و سپس از طریق یک اسکریپت پایتون به آن api ها ریکوئست زده و دیتاها را از ریپانسی که به صورت json بود استخراج کردم و در یک فایل قرار دادم.

همچنین برای این کار نیز نیاز به اسامی رستوران‌ها بود که آن دیتا را هم با api هایی که میتوانستم از اسنپ‌فود به آن‌ها ریکوئست بزنم، دریافت کردم و داده‌های مورد نیازش را برای اینکه به هر رستوران ریکوئست بزنم تا کامنت‌هایش را پیدا کنم، در آوردم.

در زیر یک نمونه را مشاهده میکنید. (ریکوئست مورد نظر برای پیدا کردن اسامی رستوران ها بود.)

```python
import requests

response = requests.get(
    "https://snappfood.ir/search/api/v1/desktop/vendors-list?lat=35.715&long=51.404&optionalClient=WEBSIT
print(response.text)
```

اما در اصل کار سخت تری در پشت این اتفاقات است، دیتا ها به این صورت به دست آمده است که ابتدا لیست تمامی رستوان ها به صورت منطقه ای گرفته شده است.

```
response = requests.get(
    "https://snappfood.ir/search/api/v1/desktop/new-home?lat

data = json.loads(response.text)
x = data["data"]["result"]
vendor_codes = set()
for i in x:
    y = i["data"]
    if "restaurants" in y:
        for j in y["restaurants"]:
            vendor_codes.add(j["vendorCode"])
```

سپس از توی این دیتا، ما به دنبال یک فیلد به اسم vendor code میگردیم که در اصل یک کد مخصوص برای هر رستوان است و سپس بر اساس vendor code ها برای هر رستوان یک ریکوئست میزنیم و کامنت ها آن را از بین ریسپانس پیدا میکنیم.

در زیر نحوه ساخت لینک برای تمامی رستوران ها آمده است.

```
my_yrl = "https://snappfood.ir/mobile/v1/restaurant/vendor-comment?client=WEBSITE&vendorCode="
my_url_2 = "&page="
my_url3 = "&sortType=score&locale=fa"

urls = []
num = 6
for i in vendor_codes:
    new_str = my_yrl + str(i) + my_url_2 + str(num) + my_url3
    urls.append(new_str)
    num += 1
    if num == 10:
        break
```

و در نهایت پیدا کردن تمامی کامنت ها:

```
for j in urls:
    response = requests.get(j)
    data = json.loads(response.text)
    x = data["data"]["comments"]
    for i in x:
        print(i["commentText"])

        print()
        for f in i["replies"]:
            print(f["commentText"])
        print()
```

استخراج داده‌های مورد نیاز از ریسپانس ریکوئست

داده هایی که از سایت اسنپ فود به دست آمده بود به صورت json بود که اطلاعات زیادی در بین آن ها بود، نمونه ای از ریسپانس خالص را در زیر مشاهده می‌کنید.

{"status":true,"data":{"count":235,"open_count":196,"finalResult":[{"type":"VENDOR","data":{"id":9346,"title":"(میدان ولیعصر) پیتزا شیلا","code":"p5

با استفاده از کتابخانه json در پایتون، دیتاهای مورد نیاز برای کامنت های یک رستوران را با تیکه کد زیر جدا کردم و آن‌ها در یک فایل ریختم.

```
print(type(data["data"]["comments"]))
x = data["data"]["comments"]

for i in x:
    print("commentText")
    print(i["commentText"])

    for j in i["replies"]:
        print("replies")
        print(j["commentText"])
    print()
```

همچنین پیدا کردن منوی رستوران ها نیز مانند اعمال بالا بود، با فرق اینکه این سری به دنبال value منو میگشتیم.

```
import requests
import json

response = requests.get("https://snappfood.ir/search/api/v1/desktop/vendors-list?lat=35.73&lo

data = json.loads(response.text)

x = data["data"]["finalResult"]
vendor_codes = set()
for i in x:
    y = i["data"]
    vendor_codes.add(y["code"])

my_url1 = "https://snappfood.ir/mobile/v2/restaurant/details/dynamic?optionalClient=WEBSITE&c
my_url2 = "&show_party=1&fetch-static-data=1&locale=fa"
f = open("New.txt", "a", encoding="utf-8")
urls = []
for i in vendor_codes:
    my_str = my_url1 + str(i) + my_url2
    urls.append(my_str)
```

```
for i in urls:
    response = requests.get(i)
    data = json.loads(response.text)
    x = data["data"]["menus"]
    for i in x:
        print(i["category"])
        f.write(i["category"])
        f.write("\n")
        for j in i["products"]:
            print(j["title"])
            f.write(j["title"])
            f.write("\n")
        print()

f.close()
```

پیش پردازش روی داده‌ها

ابتدا با استفاده از هضم یک سری پردازش داده روی متنم انجام میدهم، این پردازش ها عبارت است از:
lemmetize, stemize, tokenize , ...
با این پردازش ها بهتر میتوان کلمات را، مانند اسم، فعل، صفت و.... را پیدا کرد و بر اساس نقش کلمه آن ها را انالیز کرد.

```
def get_analysis(token, mode = 0):
    assert 0 <= mode < 5 ,"0  <= mode < 5"
    if mode == 0:
        return lemmatizer.lemmatize(token)
    if mode == 1:
        return stemmer.stem(token)
    if mode == 2:
        return normalizer.normalize(token)
    if mode == 3:
        return sent_tokenize(token)
    if mode == 4:
        return word_tokenize(token)

def ste_lem(sentence , mode = 0):
    assert 0 <= mode <= 1 ,"mode 1 or 0"
    return [get_analysis(s,1-mode) for s in sentence]

def find_NP(chunked_tree):
    res = re.findall("\[[^\[]* NP\]", chunked_tree)
    return [" ".join([get_analysis(w) for w in word[1:-4].strip().replace("\u200c", " ").split()]).strip() for word in res]

def find_VP(chunked_tree):
    res = re.findall("\[[^\[]* VP\]", chunked_tree)
    return [" ".join([get_analysis(w) for w in word[1:-4].strip().replace("\u200c", " ").split()]).strip() for word in res]
```

سپس دیتای مان را میخوانیم و آن ها را در برنامه لود میکنیم.

```
comments = []
regex = "\u200c"
comments_csv = pd.read_csv("comments.csv")["comments"]
for comment in comments_csv:
    comments.append(comment)

foods = []
foods_csv = pd.read_csv("foods.csv")["foods"]
for food in foods_csv:
    foods.append(re.sub(regex," ",food))
```

برنامه اصلی به اینگونه کار میکند که کلماتی که نقش اسم، در واقع NP و یا NE را دارند را پیدا میکند و بیشترین تعداد را برمیدارد و در نهایت کلماتی که غذا باشند را نگه میدارد و بقیه را دور میریزد.

```
number_of_each_word = {}
for comment in comments:
    tokenize_sentence = get_analysis(comment,4)
    chunked = chunker.parse(tagger.tag(tokenize_sentence))
    tree = tree2brackets(chunked)
    NP_W = find_NP(tree)
    for NP in NP_W:
        if NP not in number_of_each_word:
            number_of_each_word[NP] = 1
        else:
            number_of_each_word[NP] += 1
print(number_of_each_word)
```

نمونه ای از پردازش روی کامنت ها را در زیر میبینید:

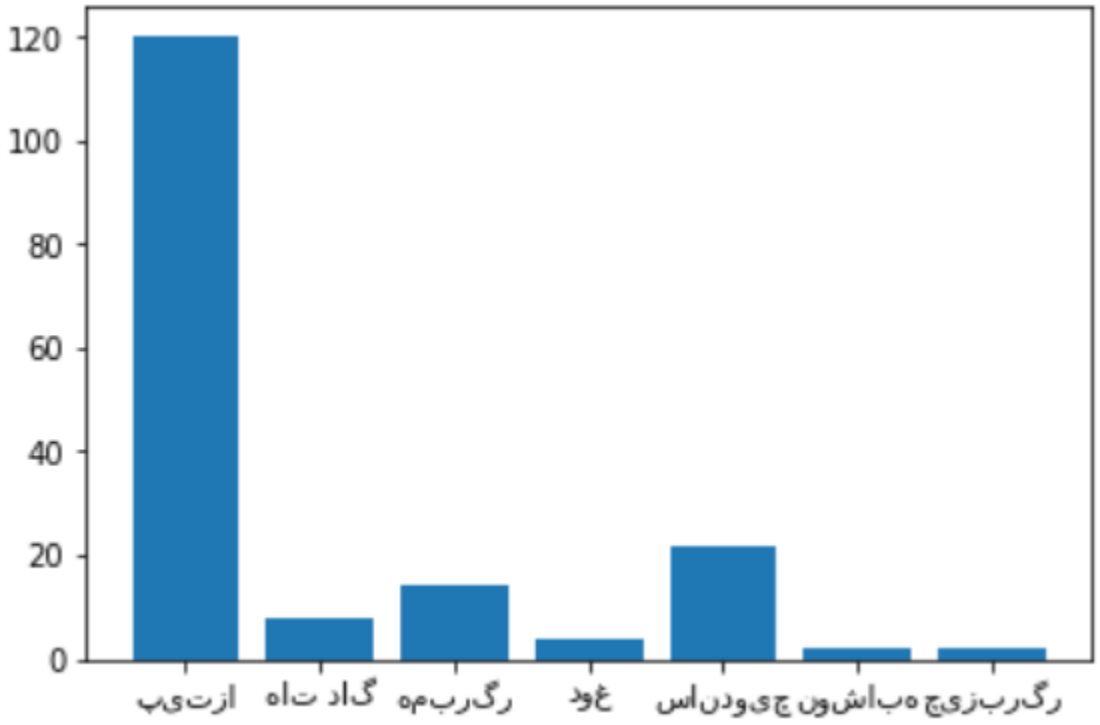
سپس بر ساس دیتابیسی از غذا ها که داریم آن ها را چک میکنیم. (این دیتابیس ممکن است تعدادی از غذا ها را نداشته باشد ک هبه صورت اختصاصی مربوط به یک رستوران هستند، مثلا پیتزا شیلا که مربوط به خود رستوران شیلا است)

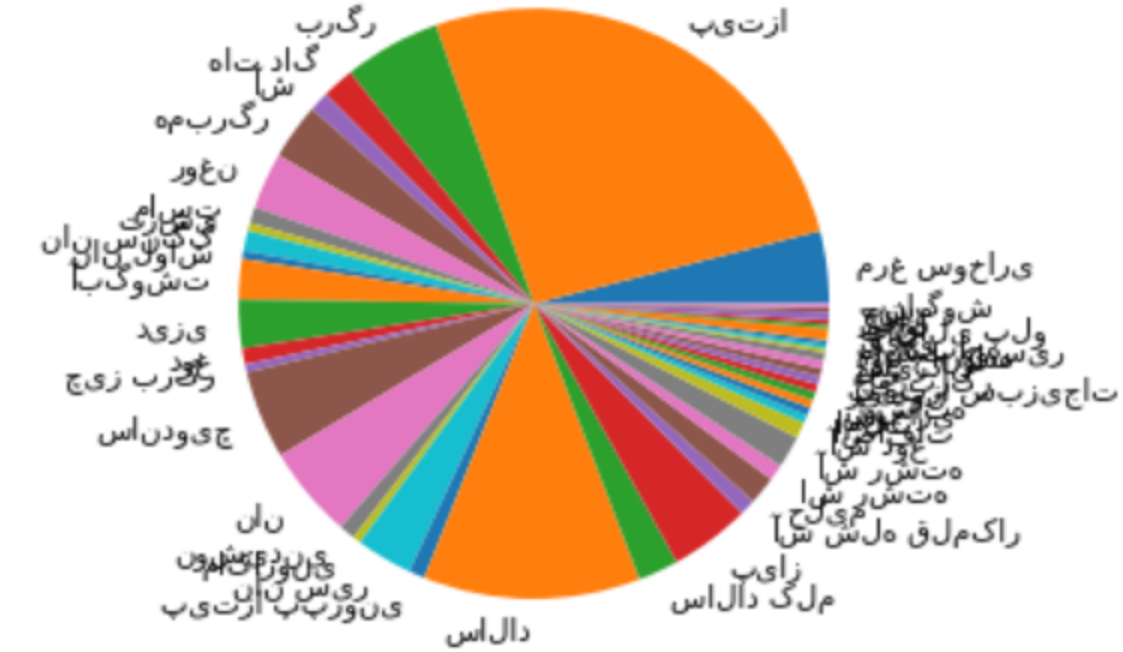
```
final_foods = {}
for word in number_of_each_word:
    if word in foods:
        final_foods[word] = number_of_each_word[word]
print(final_foods)
```

همچنین نمونه ای را در زیر بعد از پیدا کردن غذا ها میبینید:

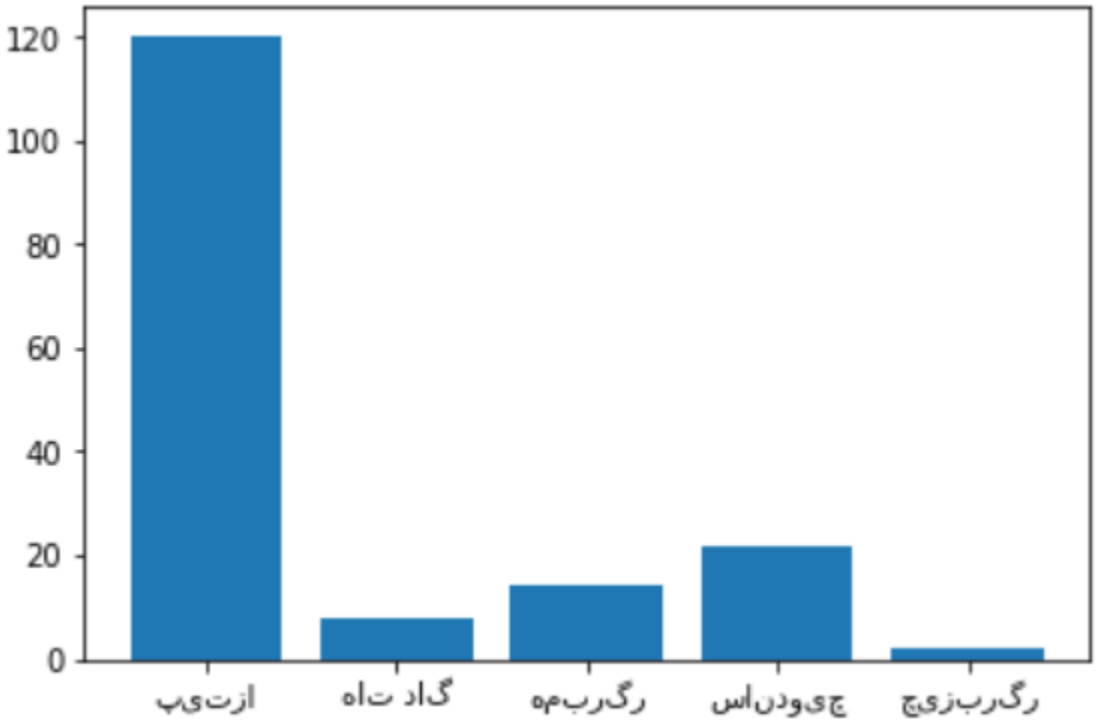
```
{'مرغ سوخاری': 72, 'پیتزا': 480, 'برگر': 96, 'هات داگ': 32, 'آش': 20, 'همبرگر': 56, 'روغن': 56, 'ماست': 16, 'ترشی': 8, 'نان سنگک': 20,
'نان لواش': 8, 'آبگوشت': 40, 'دیزی': 48, 'دوغ': 16, 'چیز برگر': 8, 'ساندویچ': 88, 'نان': 92, 'نوشیدنی': 16, 'ماکارونی': 8, 'نان سیر': 56, 'پیتز
زا پیرونی': 16, 'سالاد': 216, 'سالاد کلم': 40, 'پیاز': 80, 'آش شله قلمکار': 16, 'حلیم': 28, 'اش رشته': 16, 'آش رشته': 32, 'آش دوغ': 16, 'اضافا
ت': 8, 'املت': 8, 'سوخاری': 8, 'نوشابه': 8, 'زیتون': 8, 'پیتزا سبزیجات': 8, 'چیزبرگر': 8, 'مغز': 8, 'شیردان': 8, 'دوغ خانواده': 4, 'ماست موسیر': 4, 'کله پاچه': 4, 'لیمو': 12, 'باقالی پلو': 4, 'پاچه': 4, 'زبان': 8, 'چشم': 4, 'بناگوش': 4}
```

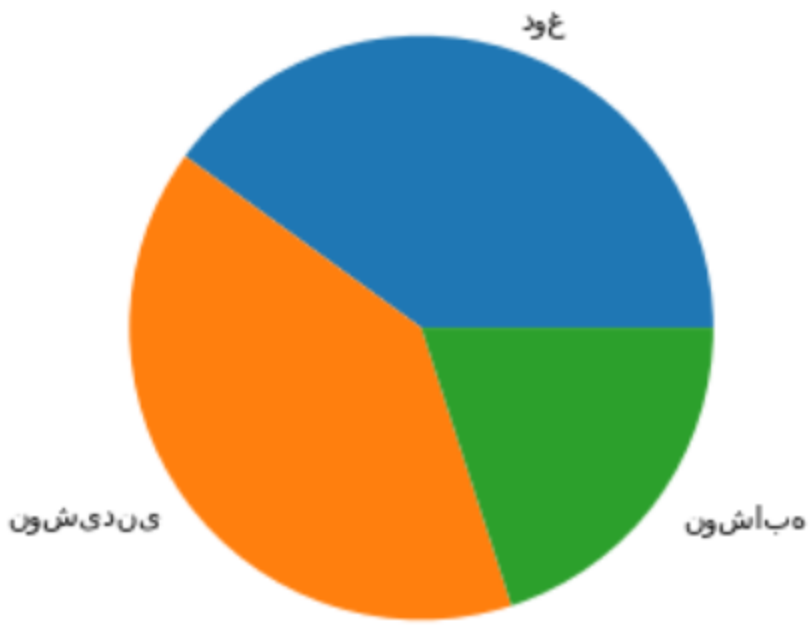
همچنین نمونه ای از تعداد را بر اساس نمودار در زیر میبینید:

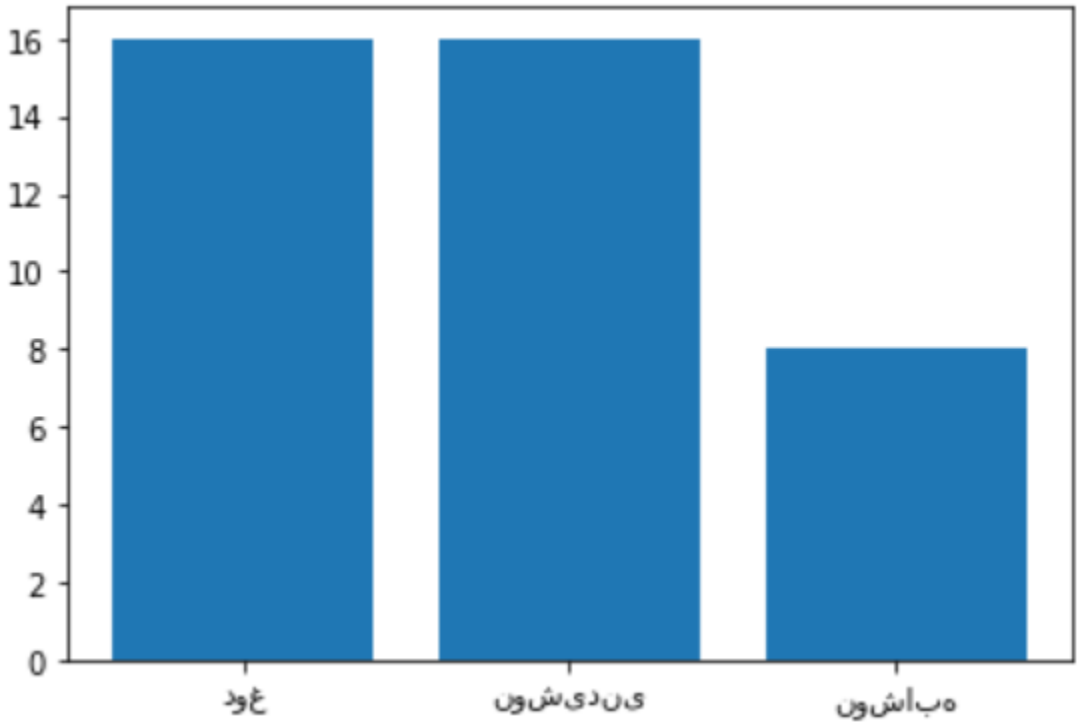




همچنین نموداری میبینید که تعداد غذا ها و نوشیدنی ها را به تفکیک نشان میدهد:




دوغ
نوشیدنی
نوشابه


16
14
12
10
8
6
4
2
0
دوغ
نوشیدنی
نوشابه