به نام خدا

گزارش پروژه شماره سوم

هوش مصنوعی

علی عدالت 810195427

دکتر صادقی

1397-1398

توضیح کلی و تعیین ویژگی های مورد استفاده

برای بررسی متن ایمیل ها در ابتدا متن ایمیل ها را نرمال می کنیم. در نرمال کردن متن ها ابتدا حروف نقطه و علامت تعجب و سوال و کاما را با اسپیس جابه جا می کنیم. دلیل این موضوع عدم اهمیت این حروف در تشخیص ایمیل هرز می باشد. در گام بعدی به تبدیل کلمات به ریشه ی خود می پردازیم که برای این کار از کتابخانه nltk استفاده می کنیم. دلیل این کار این است که جمع بودن کلمه یا زمان فعل در تشخیص ایمیل هرز در این روش تاثیری ندارد. در گام بعد نیز به حذف کلمات stopword می پردازیم. که دلیل این موضوع تکرار زیاد این کلمات است که در تشخیص کمک کننده نیست.
بعد از نرمال کردن متون به بررسی ویژگی ها بر روی دیتای تمرین می پردازیم و مدل را با توجه به نتایج بدست می آوریم.
از چهار ویژگی برای تشخیص استفاده شده است که شامل طول اولیه پیام ها در هر دو نوع و تکرار کلمات استفاده شده در هر دو نوع و وجود لینک و شماره تلفن در پیام ها می باشد. با توجه به داده ها دیده می شود که پیام های هرز طول مشخصی دارند که این موضوع در ادامه به نمودار توضیح داده شده است. همچنین با بررسی داده های هرز تمرین دیده می شود که کلماتی هستند که در این پیام ها بسیار تکرار می شوند که با بررسی تعداد تکرار کلمات می توان به این کلمات پرتکرار و تعیین کننده رسید. با بررسی داده ها دیده می شود که بیشتر پیام های هرز لینک یا شماره تلفن دارند.

الگوریتم پیاده سازی شده

در این پروژه از naive bayesian برای تشخیص پیام های هرز استفاده شده است. در naive bayesian از فرمول احتمال bayesian برای بدست آوردن معیاری که بتوان به کمک آن یک داده را کلاس بندی کرد استفاده می شود. در این پروژه برای تشخیص پیام های هرز از عبارت زیر کمک گرفته شده است.

$$p(S|E1,..,Ei)\ \alpha\ p(E1|S).\ ..\ .p(Ei|S).p(S)$$

در این عبارت وقوع ویژگی ها مستقل از هم در نظر گرفته شده است. که در این عبارت S یعنی هرز بودن پیام و Ei به معنی وجود ویژگی i ام است. پس اگر ویژگی های گفته شده در یک پیام باشد به کمک عبارت بالا احتمال هرز بودن پیام را می توان تخمین زد. برای تعیین هرز نبودن یک پیام نیز از عبارتی مشابه بالا استفاده می کنیم. در این جا برای بدست آوردن احتمال وقوع یک ویژگی به شرط هرز بودن و یا نبودن پیام از داده های تمرین استفاده می کنیم. برای بدست آوردن احتمال داشتن طول خاص یک پیام به شرط هرز بودن یا نبودن از کد زیر استفاده می کنیم.

```
def lenEvidence(self):
    hamall = 0
    spamall = 0
    for x in range(0, len(self.texts)):
        if self.trainDataTypes[x] == 'spam':
            spamall += 1
        else:
            hamall += 1
    # print((spamall, hamall))
    maxLen = len(max(self.texts, key=len))
    # print(maxLen)
    for l in range(1,maxLen):
        lenGivenSpam = 0
        lenGivenHam = 0
        for x in range(0, len(self.trainDataTypes)):
            if self.trainDataTypes[x] == 'spam' and len(self.texts[x]) == l:
                lenGivenSpam += 1
            elif self.trainDataTypes[x] == 'ham' and len(self.texts[x]) == l:
                lenGivenHam += 1
        #math.log
        if lenGivenSpam == 0:
            lenGivenSpam += 0.00005
        if lenGivenHam == 0:
            lenGivenHam += 0.00005
        self.lenProbSpam[l] = lenGivenSpam/spamall
        self.lenProbHam[l] = lenGivenHam/hamall
    self.spamall = spamall
    self.hamall = hamall
```

در این جا تعداد پیام های هرز با طول خاص L را بر تعداد کل پیام های هرز تقسیم می کنیم تا احتمال داشتن طول L به شرط هرز بودن بدست بیاید. برای بدست آوردن احتمال حضور یک کلمه خاص در یک پیام هرز یا غیر هرز از کد زیر استفاده شده است. برای بدست آوردن این احتمال ابتدا تمام متن ها را نرمال کردیم و تعداد رخ داده یک کلمه بر تمام کلمات متن های هرز را در نظر گرفتیم.

```
def vocabEvidence(self):
    i = 0
    for x in self.trainData:
        words = x.split()
        # words = list(dict.fromkeys(words))
        for y in words:
            if self.trainDataTypes[i] == 'spam':
                if not y in self.vocabFreqSpam.keys():
                    self.vocabFreqSpam[y] = 1
                else:
                    self.vocabFreqSpam[y] += 1
            else:
                if not y in self.vocabFreqHam.keys():
                    self.vocabFreqHam[y] = 1
                else:
                    self.vocabFreqHam[y] += 1
        i += 1
```

برای بدست آوردن حضور لینک در یک متن از یک regex استفاده شده است. برای بدست آوردن احتمال دیدن لینک به شرط هرز بودن یا نبودن از تابع زیر استفاده شده است.

```
def linkEvidence(self):
    i = 0
    spamGivenLink = 0
    hamGivenLink = 0
    for x in self.trainData:
        urls = re.findall('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\), ]|(?:%[0-9a-fA-F][0-9a-fA-F]))+', x)
        # print(urls)
        if self.trainDataTypes[i] == 'spam' and len(urls) > 0:
            spamGivenLink += 1
        elif self.trainDataTypes[i] == 'ham' and len(urls) > 0:
            hamGivenLink += 1
        i += 1
    if spamGivenLink == 0:
        spamGivenLink += 0.00005
    self.linkGivenSpam = float(spamGivenLink)/self.spamall
    if hamGivenLink == 0:
        hamGivenLink += 0.00005
    self.linkGivenHam = float(hamGivenLink)/self.hamall
```

برای بدست آوردن حضور شماره در یک متن از یک regex استفاده شده است. برای بدست آوردن احتمال دیدن شماره به شرط هرز بودن یا نبودن از تابع زیر استفاده شده است.

```
def phoneEvidence(self):
    i = 0
    spamGivenPhone = 0
    hamGivenPhone = 0
    for x in self.trainData:
        phones = re.findall(r'(\d{3}[-\.\s]??\d{3}[-\.\s]??\d{4}|\(\d{3}\)\s*\d{3}[-\.\s]??\d{4}|\d{3}[-\.\s]??\d{4})', x)
        # print(phones)
        if self.trainDataTypes[i] == 'spam' and len(phones) > 0:
            spamGivenPhone += 1
        elif self.trainDataTypes[i] == 'ham' and len(phones) > 0:
            hamGivenPhone += 1
        i += 1
    if spamGivenPhone == 0:
        spamGivenPhone += 0.00005
    self.phoneGivenSpam = float(spamGivenPhone)/self.spamall
    if hamGivenPhone == 0:
        hamGivenPhone += 0.00005
    self.phoneGivenHam = float(hamGivenPhone)/self.hamall
```

بعد از بدست آمدن این اطلاعات احتمال تخمینی هرز بودن پیام با وجود این ویژگی ها و احتمال تخمینی هرز نبودن پیام با وجود این ویژگی ها را به کمک تابع زیر بدست می آوریم و اگر با در نظر گرفتن این ویژگی ها احتمال هرز بودن پیام از نبودن آن بیشتر باشد آن پیام را هرز در نظر می گیریم.

```
def evaluateTest(self):
    pspam = float(self.spamall)/(self.spamall+self.hamall)
    pham = float(self.hamall)/(self.spamall+self.hamall)
    self.res = []
    spamallwords = 0
    hamallwords = 0
    for x in self.vocabFreqSpam.keys():
        spamallwords += self.vocabFreqSpam[x]
    for x in self.vocabFreqHam.keys():
        hamallwords += self.vocabFreqHam[x]
    for x in self.testData:
        spamMail = math.log(pspam)
        hamMail = math.log(pham)
        if len(x) in self.lenProbSpam.keys():
            spamMail += math.log(self.lenProbSpam[len(x)])
        else:
            spamMail += math.log(0.0000005/spamallwords)
        if len(x) in self.lenProbHam.keys():
            hamMail += math.log(self.lenProbHam[len(x)])
        else:
            hamMail += math.log(0.0000005/spamallwords)
        urls = re.findall('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+', x)
        if len(urls) > 0:
            spamMail += math.log(self.linkGivenSpam)
            hamMail += math.log(self.linkGivenHam)
        phones = re.findall(r'(\d{3}[-\.\s]??\d{3}[-\.\s]??\d{4}|\(\d{3}\)\s*\d{3}[-\.\s]??\d{4}|\d{3}[-\.\s]??\d{4})', x)
        if len(phones) > 0:
            spamMail += math.log(self.phoneGivenSpam)
            hamMail += math.log(self.phoneGivenHam)
        words = x.split()
        for y in words:
            if y in self.vocabFreqSpam.keys():
                spamMail += math.log((float(self.vocabFreqSpam[y])/spamallwords))
            else:
                spamMail += math.log(0.000005/spamallwords)
            if y in self.vocabFreqHam.keys():
                hamMail += math.log((float(self.vocabFreqHam[y])/hamallwords))
            else:
                hamMail += math.log(0.000005/hamallwords)
        if spamMail > hamMail:
            self.res.append('spam')
        else:
            self.res.append('ham')
```

بعد از بدست آوردن دیتای احتمال، تابع بالا را بر روی دیتای تست اجرا می کنیم.

Overfit

در آمار fit به این موضوع گفته می شود که که شما چقدر خوب تابع هدف را که ورودی X را به خروجی Y مپ می کند تخمین زده اید. در این جا Overfit به معنای تخمین بسیار خوب بر روی داده است. زمانی که overfit رخ می دهد مدل ما بر روی داده هایی که داریم دقت (accuracy) بالایی دارد ولی زمانی که مدل را بر روی داده های دیده نشده اجرا می کنیم دقت ما بسیار کمتر است. دلیل این موضوع وارد کردن داده های غیر کلی و نویزی در مدل می باشد. اطلاعات نویزی به صورت رندم به تشخیص کمک می کردند و در داده های غیر از داده های ما مدل وابسته به این موارد کار آمد نخواهد بود.
برای تشخیص این موضوع لازم است که مدل خود را بر روی داده دیده نشده تست کنیم که برای این کار داده های خود را به دو قسمت test و train تقسیم می کنیم و با داده های train مدل را بدست می آوریم و مدل را بر روی test آزمون می کنیم. اگر مدل در train بهتر از test باشد یعنی ما overfit داریم.

بررسی overfit در راه حل

برای این موضوع ۸۰ درصد داده را برای train جدا می کنیم و عملکرد مدل بدست آمده را بر روی تست و train بررسی می کنیم.

```
recall = 0.9552238805970149 precision = 0.9770992366412213 accuracy = 0.9911330049261083
recall = 0.9981549815498155 precision = 1.0 accuracy = 0.9997535731887629
```

در عکس بالا ردیف اول عملکرد مدل برروی تست و ردیف دوم بر روی train است که نشان می دهد overfit نداریم.

نتایج الگوریتم بر روی تست

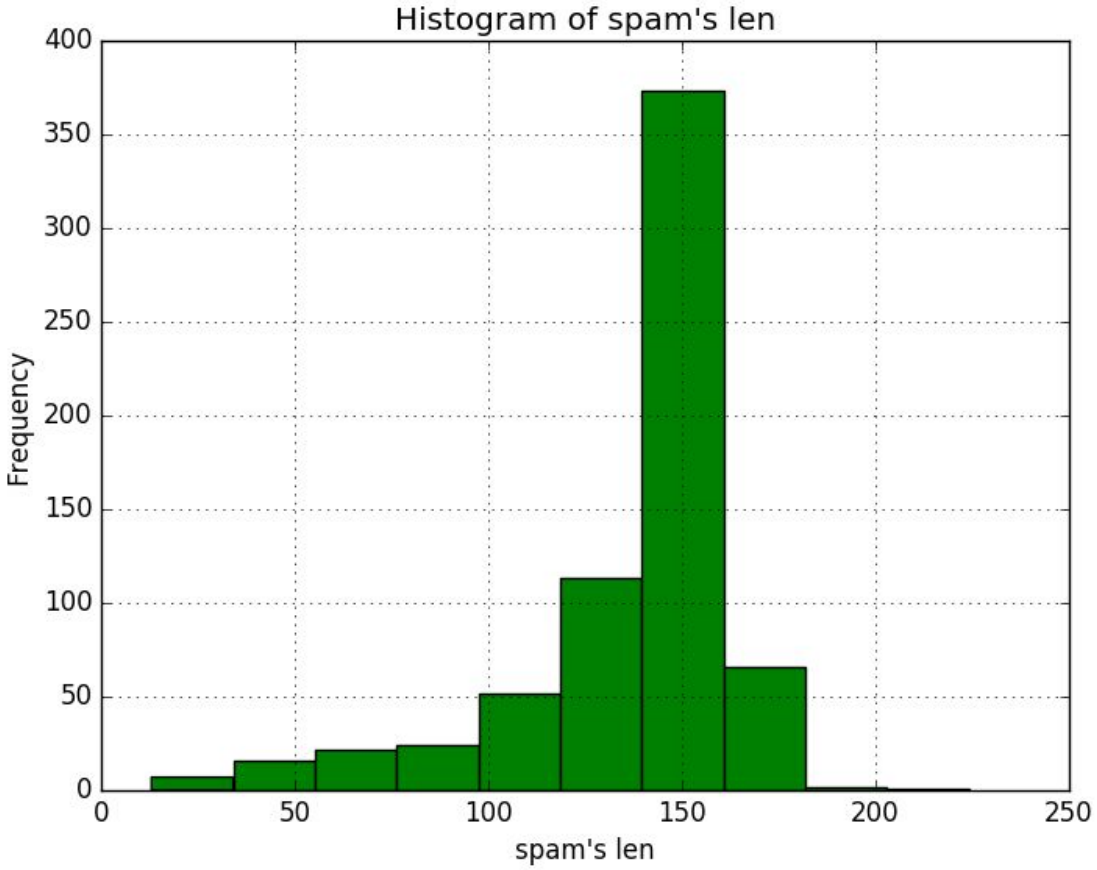
در عکس زیر مقادیر سه معیار ارزیابی خطا وقتی الگوریتم بر روی داده تست اجرا می شود آمده است. این معیارها دقت نهایی پروژه را مشخص می کند.

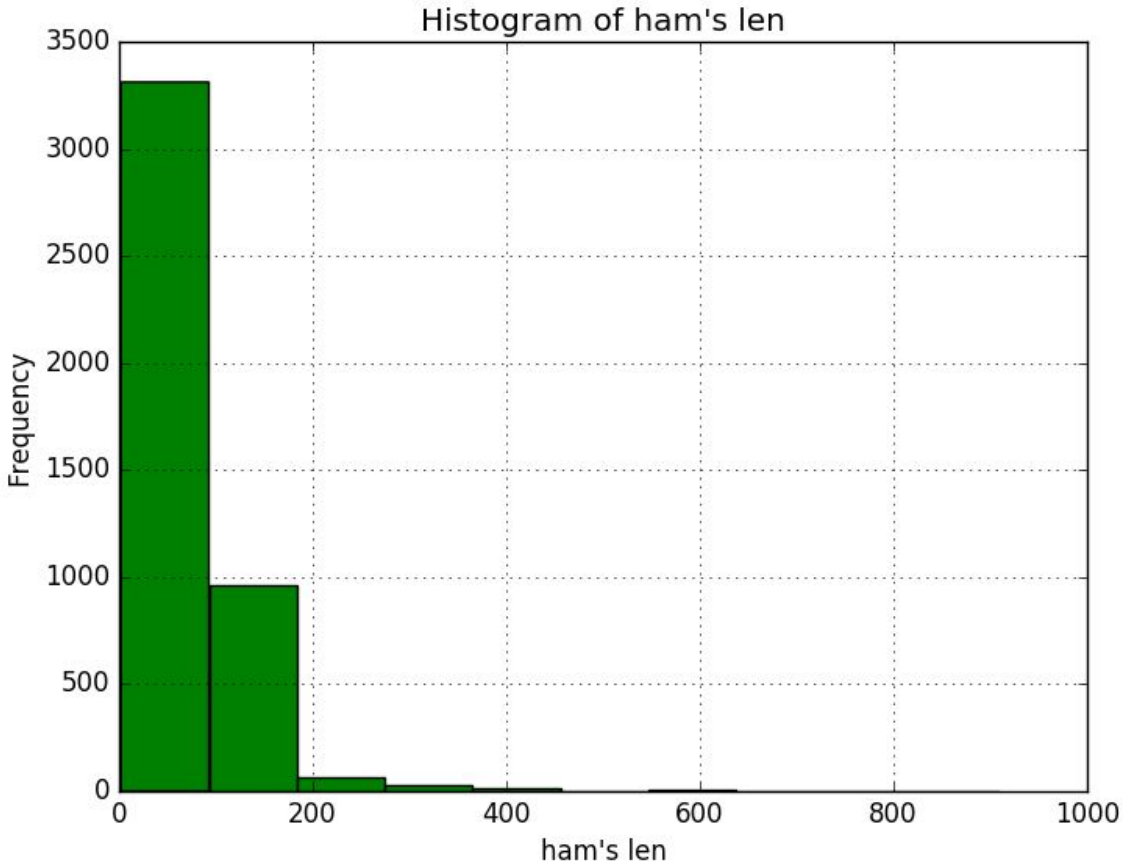
```
recall = 0.9552238805970149 precision = 0.9770992366412213 accuracy = 0.9911330049261083
```

بررسی دو ویژگی از ویژگی های در نظر گرفته شده

طول پیام

برای بررسی این که بوسیله طول پیام می توان پیام هرز را از غیر هرز تشخیص داد یا نه، نمودار فراوانی-طول پیام ها برای حالت هرز و غیر هرز را رسم می کنیم.

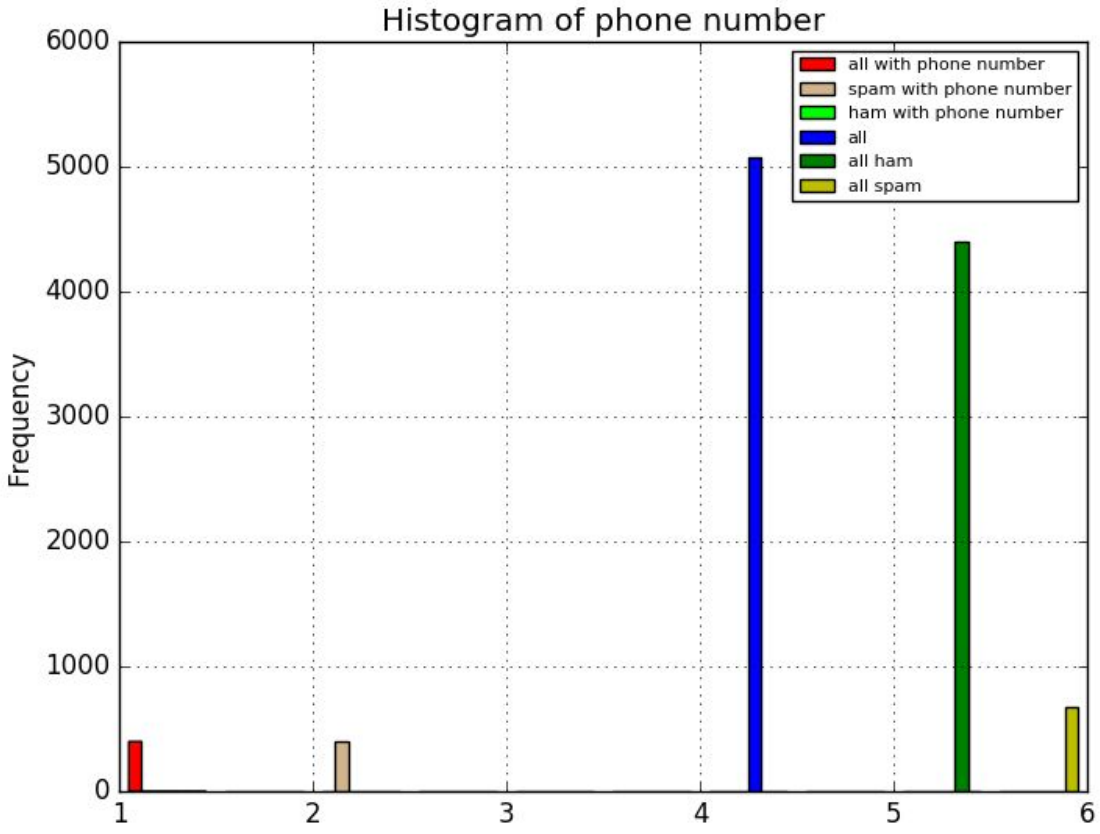




همان طور که در نمودار های بالا دیده می شود تفاوت طول پیام های هرز و غیر هرز و به طبع آن میانگین طول پیام های این دو دسته زیاد است و به همین دلیل به عنوان یک ویژگی برای تعیین کلاس پیام ها می توان استفاده کرد.

وجود شماره در پیام

برای بررسی این ویژگی نمودار زیر رسم شده است. همان طور که در نمودار دیده می شود در صد بسیار زیادی از پیام های هرز شماره تلفن دارند و در بسیار کمی از پیام های غیر هرز شماره تلفن دارند. این اختلاف زیاد به راحتی می تواند پیام ها را از نظر هرز بودن یا نبودن کلاس بندی کند. و به همین دلیل از این ویژگی استفاده شده است.


Histogram of phone number
Frequency
6000
5000
4000
3000
2000
1000
0
1
2
3
4
5
6
all with phone number
spam with phone number
ham with phone number
all
all ham
all spam