

به نام خدا



درس هوش مصنوعی و سیستم های خبره

تمرین سیزدهم-عملی

مدرس : دکتر محمدی

دانشجو : سارا سادات یونسی / ۹۸۵۳۳۰۵۳

سوالات عملی)

پیاده سازی در این تمرین قصد داریم با استفاده از فواین بیزین theorem' Bayes ی classifier پیاده سازی کنیم. برای این منظور از داده های iris استفاده کنید. ابتدا داده های خود را به صورت رندوم برهم بزنید و سپس از ۸۰ درصد آن برای آموزش و از ۲۰ درصد آن برای ارزیابی مدل استفاده کنید. در هر مرحله علاوه بر گزارش دقت کل ۲، دقت هر کلاس به صورت جداگانه نیز محاسبه شود.

بخش اول

در مرحله ی اول داده ها را بدون هیچ تغییر برای آموزش مدل استفاده کنید و نتایج را گزارش کنید .

پاسخ بخش اول)

توضیحات در نوت بوک آورده شده است.

```
Python
0.3s

ابتدا بایست چهار قسمت را تقسیم کنیم و نام کلاس را هم تقسیم کنیم چون در نهایت باید بگویم دیتا ها متعلق به کدامین کلاس هستند. برای این کار یک دیکشنری استفاده می کنیم که دیتایی که به عنوان ورودی میگیریم قسمت اخر که نام کلاس است را جدا می کنیم و به قسمت دیگر اپند می کنیم.

def Math_calculate(input):
    Total_sum = sum(input)
    Result_1 = Total_sum/float(len(input))
    return Result_1

def Dev_calculate(input):
    average = Math_calculate(input)
    var_sara = sum([(x-average)**2 for x in input])
    variance = var_sara / float(len(input)-1)
    Result_2 = math.sqrt(variance)
    return Result_2
```

```
در این قسمت با استفاده از توابع آماده و عتوابعی که نوشتم سعی می کنیم فرمول محاسبه ی گوسین را پیاده سازی کنیم. به ترتیب میانگین و انحراف معیار و نوع کلاس را مشخص می کنیم و مقدار احتمال گوسین را محاسبه می کنیم.

def classes_Gaussian_PDF(Math_total, given_part):
    summation_R = sum([Math_total[label][0][2] for label in Math_total])
    Prior_probability = dict()
    for class_Select_2, class_Math_total in Math_total.items():
        Prior_probability[class_Select_2] = Math_total[class_Select_2][0][2] / \
            float(summation_R)
    for i in range(len(class_Math_total)):
        Math_calculate, Dev_calculate, count = class_Math_total[i]
        Prior_probability[class_Select_2] *= Gaussian_PDF(given_part[i],
            Math_calculate, Dev_calculate)
    return Prior_probability
```

سوال یک و دو

همانگونه که سوال خواسته است داده هارا بر می زنیم و به صورت 80 درصد برای آموزش و 20 درصد جهت تست گرفتن جدا می کنیم که اگر مقدار داده برای آموزش در این قسمت بیشتر شود دقت آن بیش تر می شود.

```
iris = open("../iris.data")
iris = list(iris)
input = []
w= len(iris)-1
for i in range(len(iris)):
    if i == w:
        break
    given_part = iris[i].split(",")
    given_part[-1] = given_part[-1][::-1]
    given_part[0] = float(given_part[0])
    given_part[1] = float(given_part[1])
    given_part[2] = float(given_part[2])
    given_part[3] = float(given_part[3])
    input.append(given_part)

train = random.sample(input, k=round(len(input) * 0.8))
test = []
for i in range(len(input)):
    if input[i] not in train:
        test.append(input[i])

Bys(train, test)
```

```
y_pred = Y.fit(X_train, y_train).predict(X_test)
print("Question_3")
print("Accuracy Scikit : ", 100*(y_test == y_pred).sum()/X_test.shape[0])
print("*****")
```

[6] ✓ 0.4s

```
... *****
Question_1
Total Accuracy 92.85714285714286
*****
Accuracy Iris-virginica 75.0
Accuracy Iris-setosa 100.0
Accuracy Iris-versicolor 100.0
*****
Question_3
Accuracy Scikit : 93.33333333333333
*****
```

بخش دوم (امتیازی)

سعی کنید با تغییراتی که روی داده ها اعمال می کنید نتایج را بهبود بدهید و جزئیات آن را مرحله به مرحله در گزارش خود ذکر کنید. در صورتی که تغییرات اعمال شده به بهبود مدل کم نکرد، آن تغییرات را هم در گزارش خودتان بنویسید .

به ترتیب دقت هارا با در کد مساوی ۸۰ و ۹۰ و ۹۵ گذاشتیم و میبینم که دقت افزایش پیدا می کند.

پاسخ بخش دوم)

در این نوع طبقه بندی اختصاص دادن داده های تست و آموزش بسیار اهمیت دارد به طوری که اگر درصد داده آموزشی را بالاتر ببریم به مراتب دقت برنامه و accuracy افزایش می یابد.

برای مثال اگر داده های آموزشی را از ۸۰ درصد به ۹۰ درصد و داده های آزمایشی از ۲۰ به ۱۰ برسد خواهیم داشت :

```
train = random.sample(input, k=round(len(input) * 0.9))
test = []
for i in range(len(input)):
    if input[i] not in train:
        test.append(input[i])
```

MS OUTPUT DEBUG CONSOLE TERMINAL

R: VARIABLES

Name ▲ Type
ables defined

TERMINAL

```
PS C:\Users\user\Desktop\AI 13> python AB.py
*****
Question_1
Total Accuracy 93.33333333333333
*****
Accuracy Iris-setosa 100.0
Accuracy Iris-versicolor 100.0
Accuracy Iris-virginica 75.0
*****
Question_3
Accuracy Scikit : 94.81481481481481
*****
```

```
X, y = datasets.load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.95, random_state=0)
Y = GaussianNB()
y_pred = Y.fit(X_train, y_train).predict(X_test)
print("Question_3")
print("Accuracy Scikit : ", 100*(y_test == y_pred).sum()/X_test.shape[0])
print("*****")
```

MS OUTPUT DEBUG CONSOLE TERMINAL

R: VARIABLES

Name ▲ Type
ables defined

TERMINAL

```
PS C:\Users\user\Desktop\AI 13> python AB.py
*****
Question_1
Total Accuracy 100.0
*****
Accuracy Iris-versicolor 100.0
Accuracy Iris-virginica 100.0
Accuracy Iris-setosa 100.0
*****
Question_3
Accuracy Scikit : 65.73426573426573
*****
PS C:\Users\user\Desktop\AI 13> 
```

وقتی اینجا به ۹۵ صدم داده های آموزشی را رساندیم دقت برابر ۱۰۰ شد !

بخش سوم

در این بخش نتایج بدست آمده از پیاده سازی خودتان را با نتایج توابع آماده learn-scikit مقایسه کنید.

پاسخ بخش سوم)

با بررسی این دو مورد به این نتیجه میرسیم که میانگین دقت دو روش به هم نزدیک است و هر دو آن ها دقت مناسبی در پیش بینی موارد مختلف ورودی دارند و روش پیاده سازی شده توسط ما هم از دقت و صحت خوبی برخوردار می باشد. و تست با دقت بالایی کلاس بندی می شود.

و از آنجایی که متغیر ها به صورت پیوسته اند روش گاوسی انتخابی بهترین روش میان این روش هاست.

```
X, y = datasets.load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.8, random_state=0)
Y = GaussianNB()
y_pred = Y.fit(X_train, y_train).predict(X_test)
print("Question_3")
print("Accuracy Scikit : ", 100*(y_test == y_pred).sum()/X_test.shape[0])
print("*****")
```

```
y_pred = Y.fit(X_train, y_train).predict(X_test)
print("Question_3")
print("Accuracy Scikit : ", 100*(y_test == y_pred).sum()/X_test.shape[0])
print("*****")
```

[6] ✓ 0.4s

```
... *****
Question_1
Total Accuracy 92.85714285714286
*****
Accuracy Iris-virginica 75.0
Accuracy Iris-setosa 100.0
Accuracy Iris-versicolor 100.0
*****
Question_3
Accuracy Scikit : 93.33333333333333
*****
```