تمرین سوم ,bayes

یادگیری ماشین

پویا شریفی 9823117

Naïve bayes:

این کلمه از دو بخش تشکیل شده است ,یکی نایو و یکی بیز:

بیز که همین فرمول است:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

و به پیش فرض های ما هم نایبو گفته میشوند ,چون پیش فرض ما این است که تمام فیچر های ما از هم independent هستند که این در بیشتر مواقع غلط است.

ما در این تمرین از لایبرری naïve bayes استفاده میکنیم :

NB in sklearn

In sklearn there are 3 main functions you can use to perform Naive Bayes:

- GaussianNB(): Assumes that features follow a Normal/Gaussian Distribution.
- BernoulliNB(): Assumes features are binary (0/1)
- CategoricalNB(): Assumes features are discrete categories (can have more than 2 categories)

This means that if your features are continuous you'd use GaussianNB(), if they are only binary, use BernoulliNB() and if they are only Categorical, use CategoricalNB(). In practice, we'll often use either GaussianNB() or CategoricalNB() (since CategoricalNB() can also handle it when we have binary + categorical).

This means that computationally, we cannot have both continuous + categorical predictors in one sklearn NB model. (There are workarounds for this: see here, but for now, we'll be using only one or the other).

حالا سراغ تمرین میرویم:

هدف ما از این تمرین این است که طبقه بندی افراد با سرطان است.

اول library های مد نظر را لود میکنیم:

```
from sklearn.naive_bayes import GaussianNB ,BernoulliNB, CategoricalNB
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.metrics import accuracy_score
from sklearn.metrics import completeness_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
```

اول دیتا را لود میکنیم:

```
df=pd.read_csv("BreastCancer.csv")
```

حالا بخش data و lable را جدا میکنیم و ستون sample code number را حذف میکنیم:

```
x=df.iloc[:,1:10]
y=df.iloc[:,10]
```

حالا بخش train و test را جدا میکنیم:

```
kf= KFold(n_splits=5)
```

```
for train,test in kf.split(x):
    X_train=x.iloc[train]
    # print(X_train)
    y_train=y.iloc[train]
    X_test=x.iloc[test]
    y_test=y.iloc[test]
```

از تابع multinomial استفاده میکنیم:

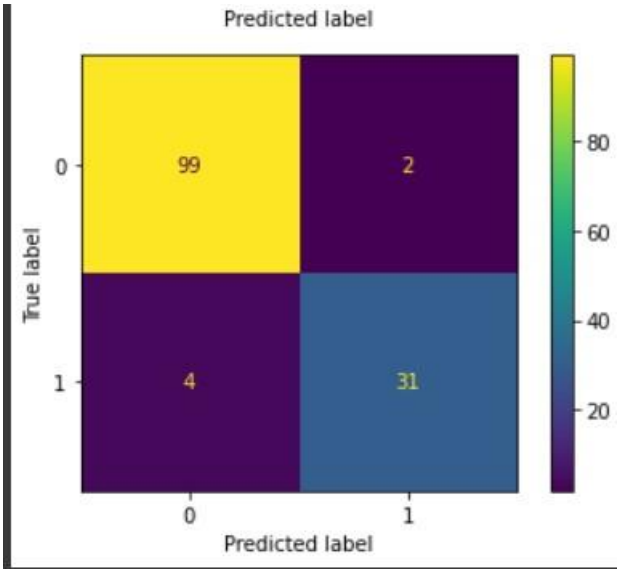
```
nb=MultinomialNB()
```

حالا fit میکنیم و برای پیدا کردن accuracy از کد زیر استفاده میکنیم:

```
acc_train.append(accuracy_score(y_train,nb.predict(X_train)))
    acc_test.append(accuracy_score(y_test,nb.predict(X_test)))
```

حالا برای visualize کردن بهتر output مون از confusion matrix استفاده میکنیم:

```
plot_confusion_matrix(nb,X_test,y_test)
```



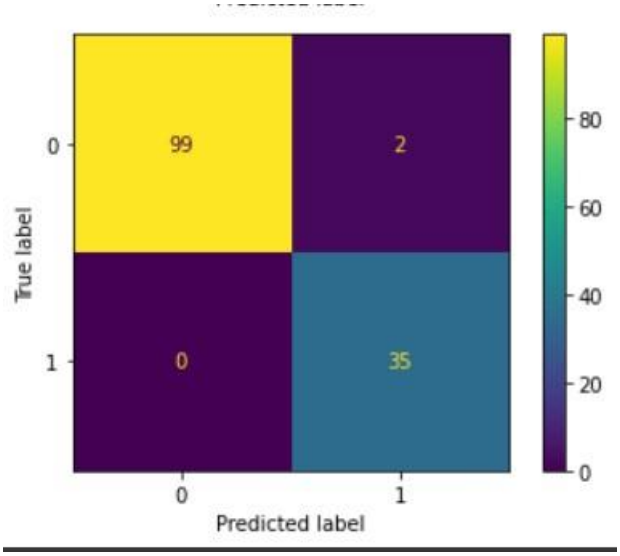
و accuracy آن:

0.893237440961786

0.9059351373793787

بالایی میانگین accuracy تست و پایینی میانگین ترین

حالا برای بخش ب) هم همینکارو تکرار میکنیم ,فقط با گوسین



0.9605195362816659

0.9637650588290441

همان طور که میبینید ,0.07 درصد دقیق تر از مدل قبل است,و میبینید که false negative هم نداریم.