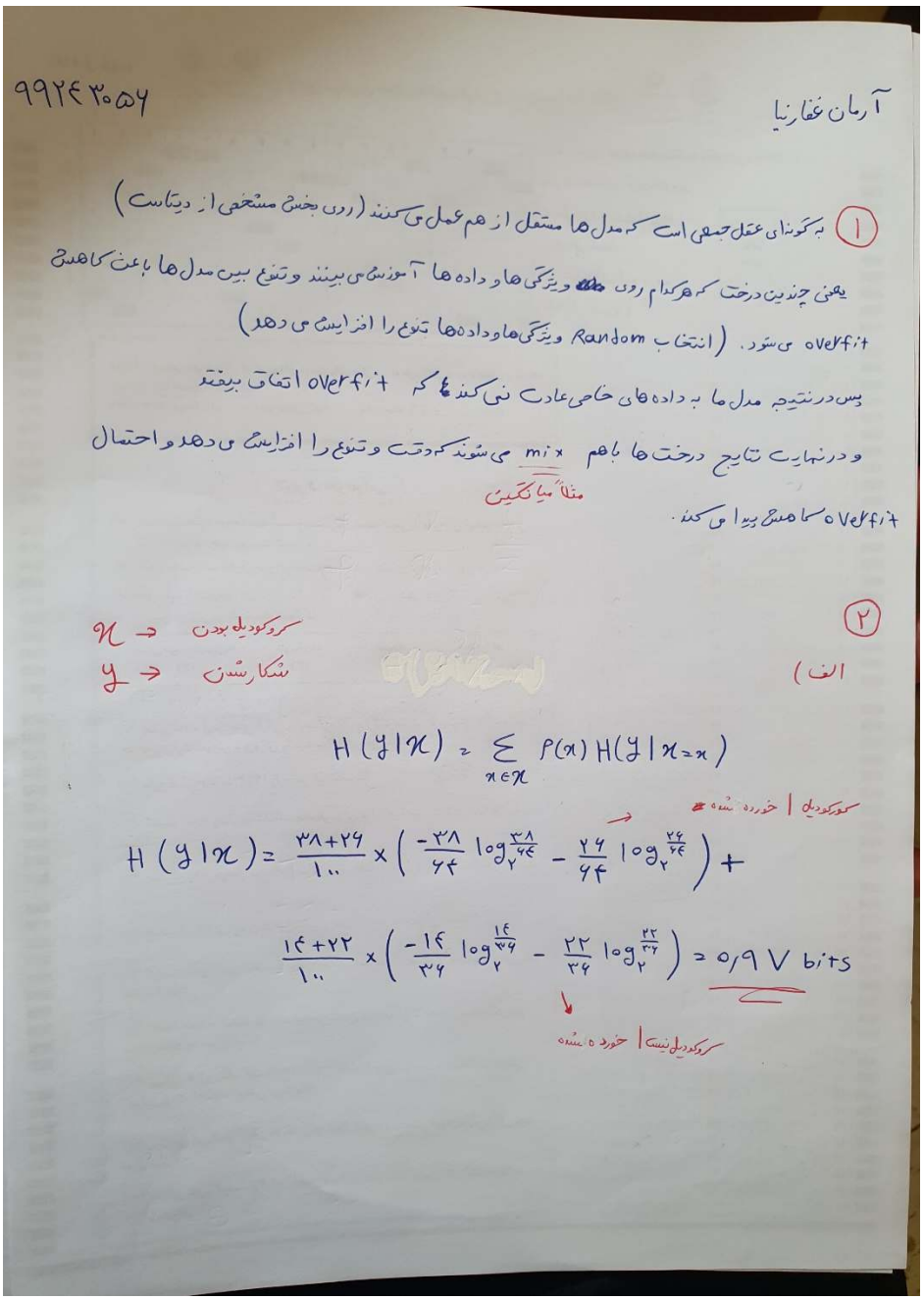
گزارش تمرین اول

آرمان غفارنیا 99243056

تحلیلی :



ب)

$$IG(Y|X) = H(Y) - H(Y|X)$$

$$IG(Y|X) = -\left(-\frac{\text{۵۲}}{\text{۱۰۰}}\log_2\frac{\text{۵۲}}{\text{۱۰۰}} - \frac{\text{۴۸}}{\text{۱۰۰}}\log_2\frac{\text{۴۸}}{\text{۱۰۰}}\right) - 0,97 = 0,003 \text{ bits}$$

خورده‌شده و خورده‌نشده

ج)

$$H(Y|X) = \sum_{x \in \mathcal{X}} P(x)\, H(Y|X=x)$$

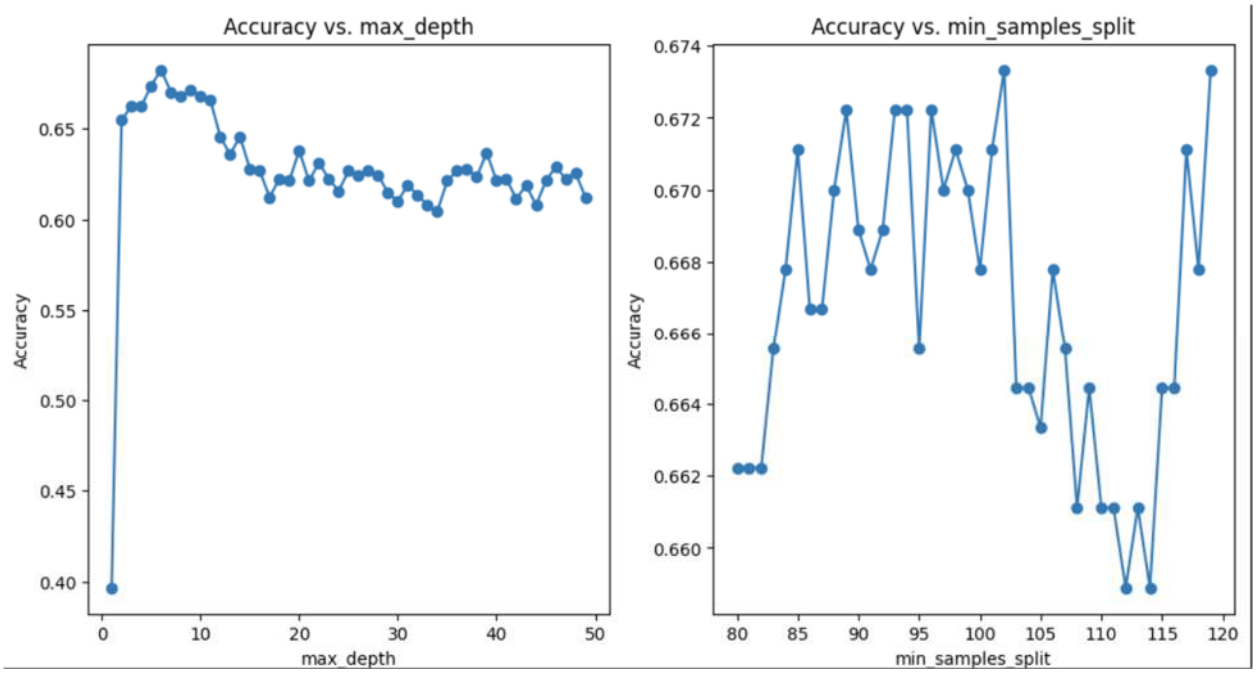
$$-\frac{\text{۲۶}}{\text{۴۸}}\log_2\frac{\text{۲۶}}{\text{۴۸}} - \frac{\text{۲۲}}{\text{۴۸}}\log_2\frac{\text{۲۲}}{\text{۴۸}} \simeq 0,99 \text{ bits}$$

خورده‌شده | کوروکودیل است — خورده‌نشده | کوروکودیل نیست

کدی :

-1

ب) کد درخت تصمیم زده شد. همچنین بررسی برای رسیدن به دقت بالا به کمک 2 ابرپارامتر انجام شد. (max_depth , min sample split) یکی به معنای بیشترین عمق ممکن درخت و دیگری یعنی در هر نود اگر نمونه ها از چه تعدادی کمتر باشد دیگر 2 شاخه نمیشود.



بررسی روی این 2 ابرپارامتر نشان می دهد که بهترین دقت در موقعی به دست می آید که ماکسیمم عمق درخت در حدود 6 و min_sample_split در حدود 102 باشد.

انتخاب این 2 مقدار برای درخت میتواند منجر به بهبود دقت مدل ما شود.

ج)

رندوم فارست و گرادیانت بوستینگ آموزش داده شد و دقت هم دریافت شد.

رندوم فارست : دقت 75 درصد به دست آمد با 10 درصد داده ها به عنوان تست.

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size= 0.1)
R_Forest = RandomForestClassifier()          #create instance of random-fores
R_Forest.fit(x_train, y_train)               # train the random-forest
accuracy = R_Forest.score(x_test, y_test)    #count accuracy
print(accuracy)
```

```
0.7511111111111111
```

گرادیانت بوستینگ : دقت 72 درصد به دست آمد با 10 درصد داده ها به عنوان تست.

```
[4] x_train, x_test, y_train, y_test = train_test_split(x, y, test_size= 0.1)
    Gradient = GradientBoostingClassifier()                                  #create
    Gradient.fit(x_train, y_train)                                           #train t
    accuracy = Gradient.score(x_test, y_test)                                #count a
    print(accuracy)
```

```
0.72
```

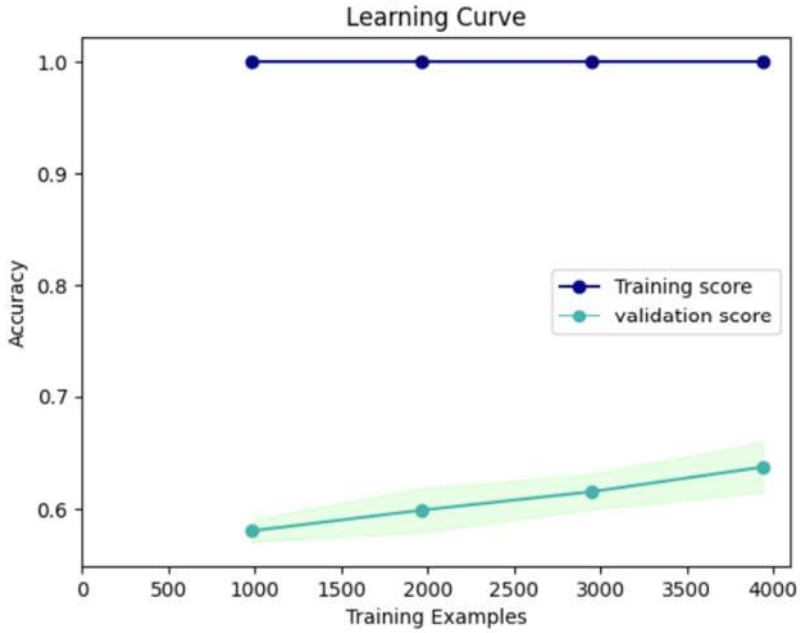
بخش learning curve :

این نمودار برای نشان دادن این است که در مدل ما با افزایش داده training عملکرد مدل ما بهتر میشود.

CV عددی است که نشان میدهد train چندبار انجام شده مثلا عدد ما 8 به این معناست که مدل ما 8 بار train داده شده است. به این معنا که داده مربوط به بخش training را 8 بخش مختلف را و هر بار یکی از آنها را امتحان میکند و نتایج را باهم میانگین میگیرد.

خروجی نمودار :

درخت تصمیم :



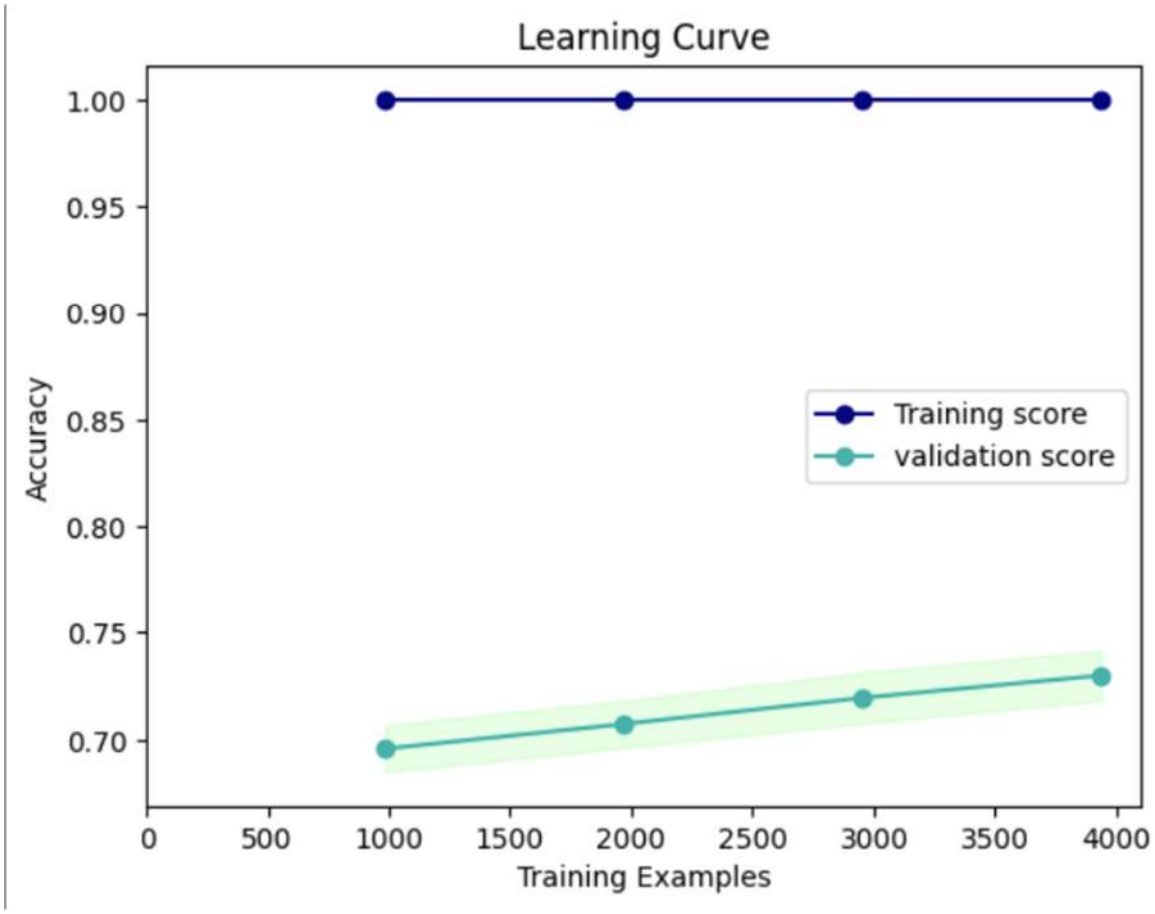
یک خط نمودار نتیجه عملکرد مدل روی همان training set است که قطعا دقت 100 درصد دارد. (عمق درخت نامحدود است و کاملا روی دیتا fit میشود)

خط دیگر نتیجه عملکرد مدل روی validation است که در نتیجه افزایش تعداد داده های training دقت بهتری پیدا میکند و به مدل ایده آل نزدیک میشود.

برای نمایش بهتر از انحراف معیار هم استفاده میکنیم و به شکل یک هاله اطراف خط دقت نمایش میدهیم.

انحراف از معیار در خط بالا به دلیل fit شدن کامل وجود ندارد.

رندوم فارست :



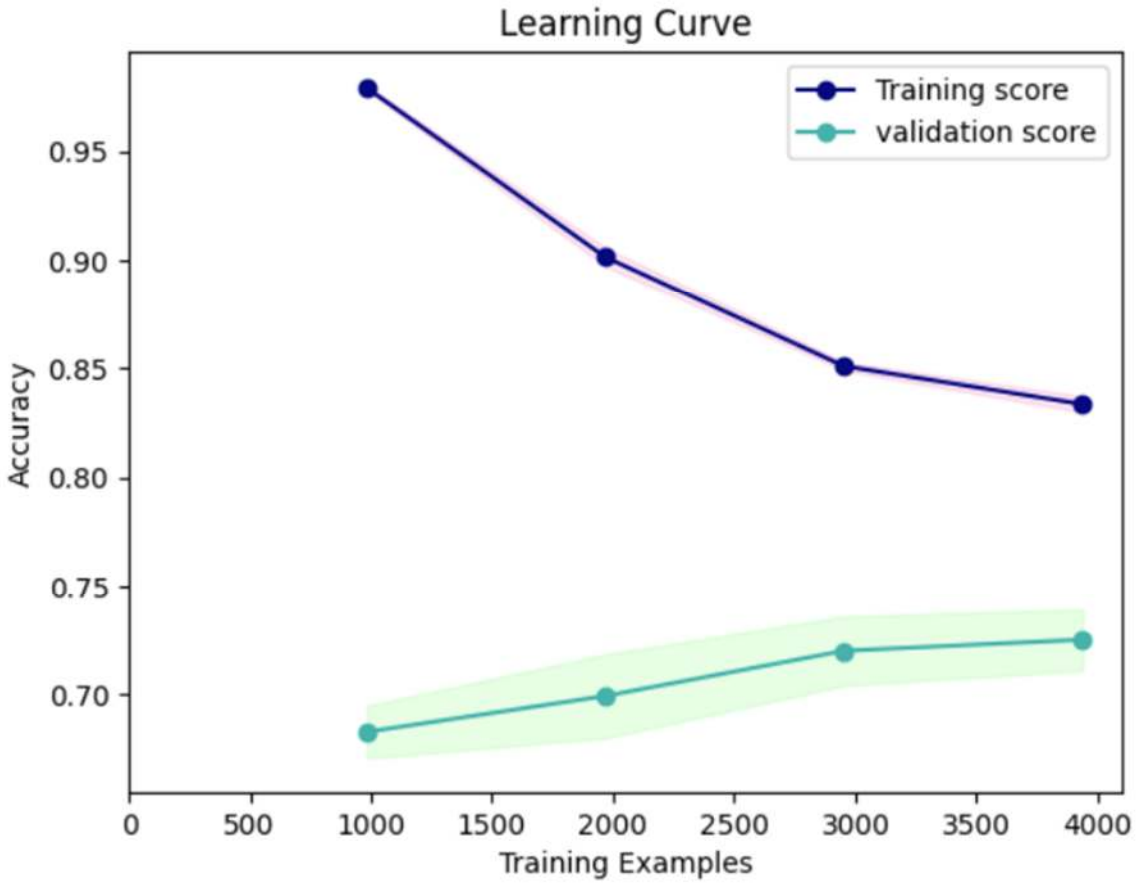
یک خط نمودار نتیجه عملکرد مدل روی همان training set است که قطعا دقت 100 درصد دارد. (عمق درخت نامحدود است و کاملا روی دیتا fit میشود)

خط دیگر نتیجه عملکرد مدل روی validation است که در نتیجه افزایش تعداد داده های training دقت بهتری پیدا میکند و به مدل ایده آل نزدیک میشود.

برای نمایش بهتر از انحراف معیار هم استفاده میکنیم و به شکل یک هاله اطراف خط دقت نمایش میدهیم.

انحراف از معیار در خط بالا به دلیل fit شدن کامل وجود ندارد.

گرادیانت بوستینگ :



تحلیل این نمودار مثل موارد بالا است با این تفاوت که در موارد بالا درخت تا هر عمقی که میخواست میتوانست ساخته شود و روی داده train کاملا فیت شود و موقع تست روی همین داده ها دقت 100 بدهد ولی گرادیانت عمق نامحدود ندارد به همین دلیل همواره دقت 100 ندارد .

خط پایین با افزایش تعداد داده train عملکرد بهتری پیدا میکند و دقت اش افزایش میابد و به نمودار بالا نزدیک تر میشود.

انحراف از معیار در خط بالا بوجود آمد چون دیگر مدل نتوانسته روی داده train فیت شود.

-2

الف) پیاده سازی در فایل knn انجام شده است

```
[33] accuracy = accuracy_score(y_test, y_pred)        #find accuracy
     print(accuracy)

     0.7714285714285715
```

دقت 77 درصد گرفتیم. (با تابع accuracy_score)

ب) نمودار نسبت TPR به FPR است

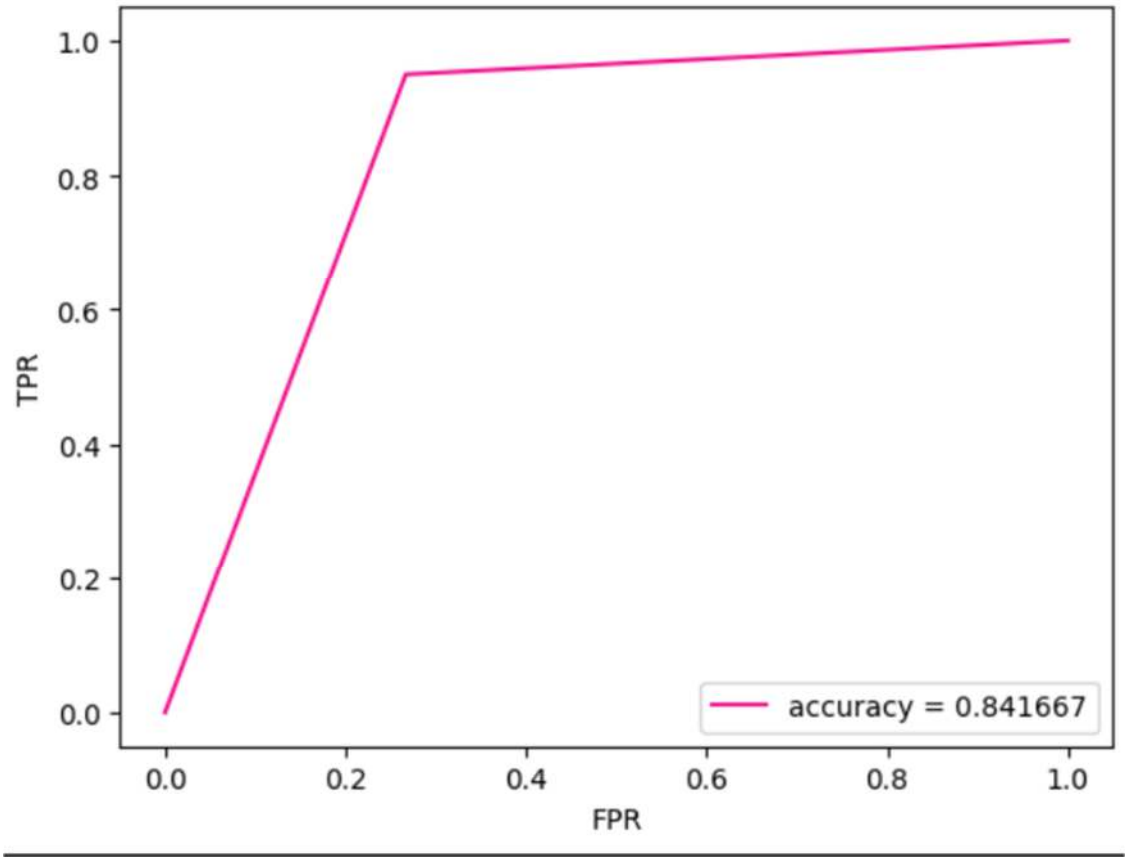
در حقیقت نشان میدهد که چقدر کلاس positive را درست تشخیص داده ایم به نسبت کل داده های positive در مقابل داده هایی که negative هستند و ما positive تشخیص داده ایم نسبت به کل داده های negative .

همچنین مساحت زیر نمودار به ما دقت مدل را نشان میدهد

کاربرد :

1- برای مقایسه مدل ها و اینکه کدام مدل دقت و عملکرد بهتری دارد و برای استفاده ما مناسب تر است
2- همچنین با نشان دادن TPR , FPR به ما کمک میکندکه در جهت بهبود مدل حد آستانه را تغییر دهیم و با مقادیری مدل را پیاده سازی کنیم که عملکرد بهتری داشته باشد (تنظیم k و تعداد داده های تست و train)

3- با مقایسه نمودار با خط y =x کمک میکند بفهمیم آیا مدل ما از حالت 50 /50 بهتر عمل کرده یا نه و آیا مدل خوب و کاربردی است یا خیر



نمودار بالای خط y =x قرار گرفته است و شبیه نمودار رادیکالی است این به این معناست که عملکرد بهتری به نسبت حالت 50/50 دارد و درکل شکل نمودار نشان دهنده عملکرد قابل قبول الگوریتم است و همچنین دقت حدود 84 درصد دارد یعنی مساحت زیر نمودار 0.84 است .

```
[45] accuracy = accuracy_score(y_test, y_pred)          #find accuracy
     print(accuracy)

     0.8571428571428571
```