**باسمه تعالی**

**تکلیف سری چهارم داده کاوی**

**سارا برادران (شماره دانشجویی : ۹۶۲۴۱۹۳)**

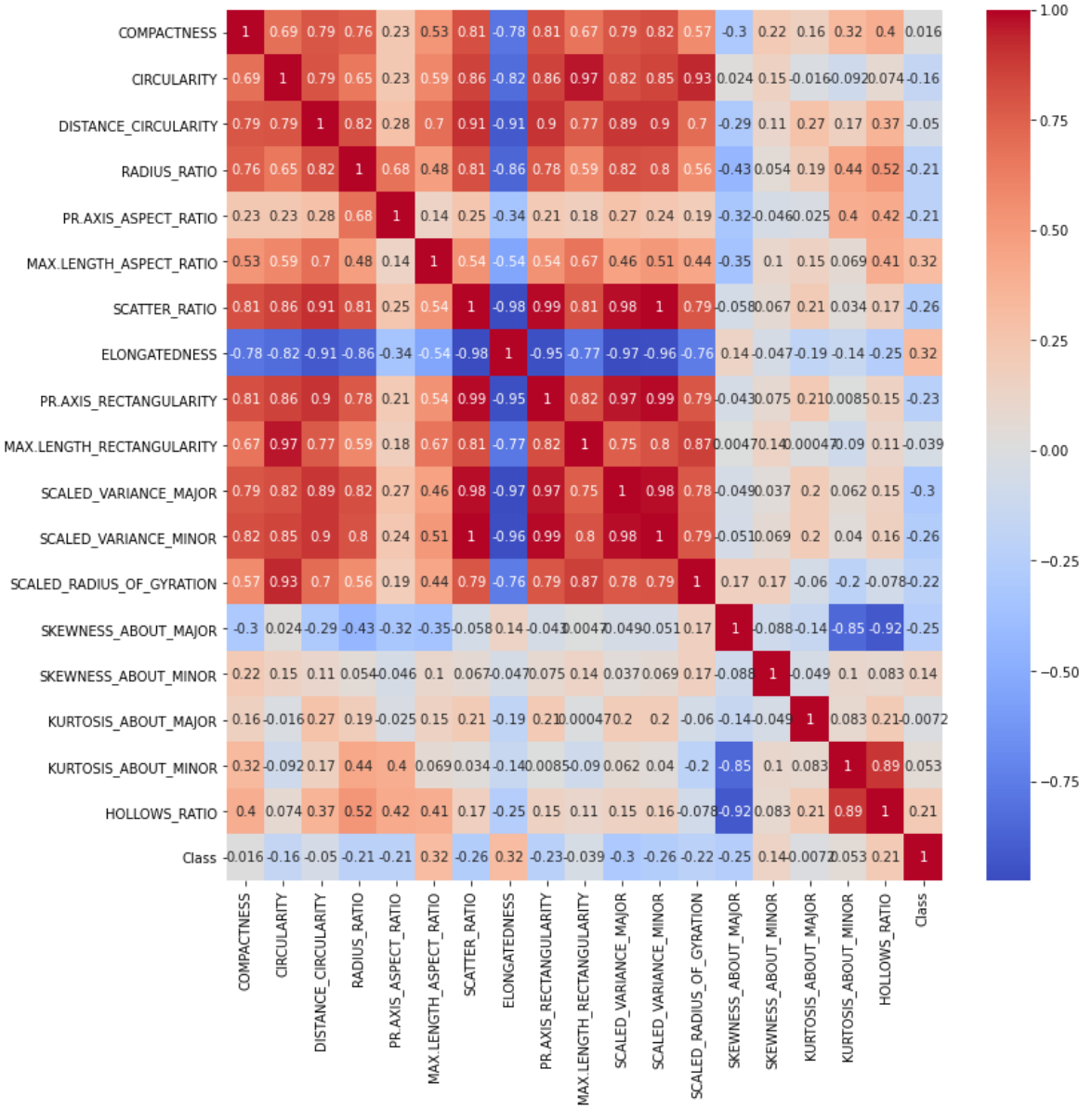
---

سوال ۱ )

a) با استفاده از متد read_csv دیتاست را خوانده و به دیتافریم تبدیل می کنیم.

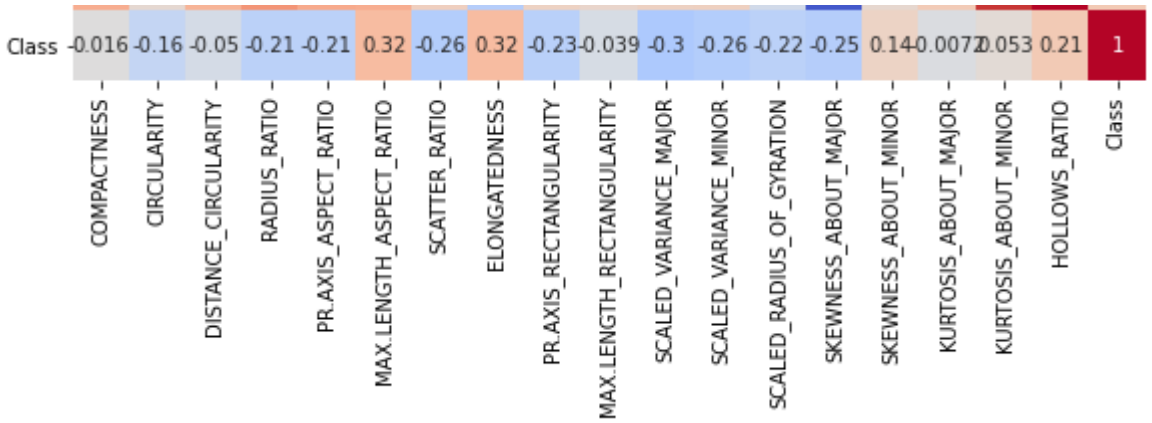
b) ابتدا تمام missing value ها جستجو و با nan جایگذاری شده است. (البته به نظر دیتاست مربوطه فاقد missing value بوده است) سپس با استفاده از روش z_score داده های پرت هر ستون را حذف کرده و نهایتا پس از encode کردن ستون های دسته ای (فیلد class) از دیتافریم ایجاد شده برای انجام پردازش ها در قسمت بعدی استفاده می نماییم.

c) نمودار heatmap به صورت زیر رسم شده است :



تفسیر نمودار : با توجه به اینکه برخی از ستون های دیتاست correlation نسبتا زیادی با یکدیگر دارند لذا می توان برخی فیلد ها را نماینده برخی دیگر در نظر گرفته و به اصطلاح dimention_reduction صورت دهیم. برای مثال فیلد PR.AXIS_RECTANGULARITY و SCATTER_RATIO دارای میزان correlation حدودا 0.99 هستند و یا دو فیلد SCATTER_RATIO و SCALED_VARIANCE_MINOR که داری میزان correlation برابر ۱ هستند. لذا برای مثال می توان در حالت کلی فیلد SCATTER_RATIO را نماینده دو فیلد دیگر در نظر گرفت.

ستون های موثر در فیلد class را با استفاده از این نمودار نمی توان بدست آورد چرا که هیچ از فیلد های داده شده به تنهای correlation چندانی با فیلد Class نداشته اند لذا احتمالا از محاسبات بر روی چندین فیلد می بایست فیچر جدیدی ایجاد کرد که ارتباط نسبتا قوی با فیلد Class داشته باشد.




(d

```
# 1-d
y = df_zscore[['Class']]
x = df_zscore.drop(['Class'], axis=1)
```

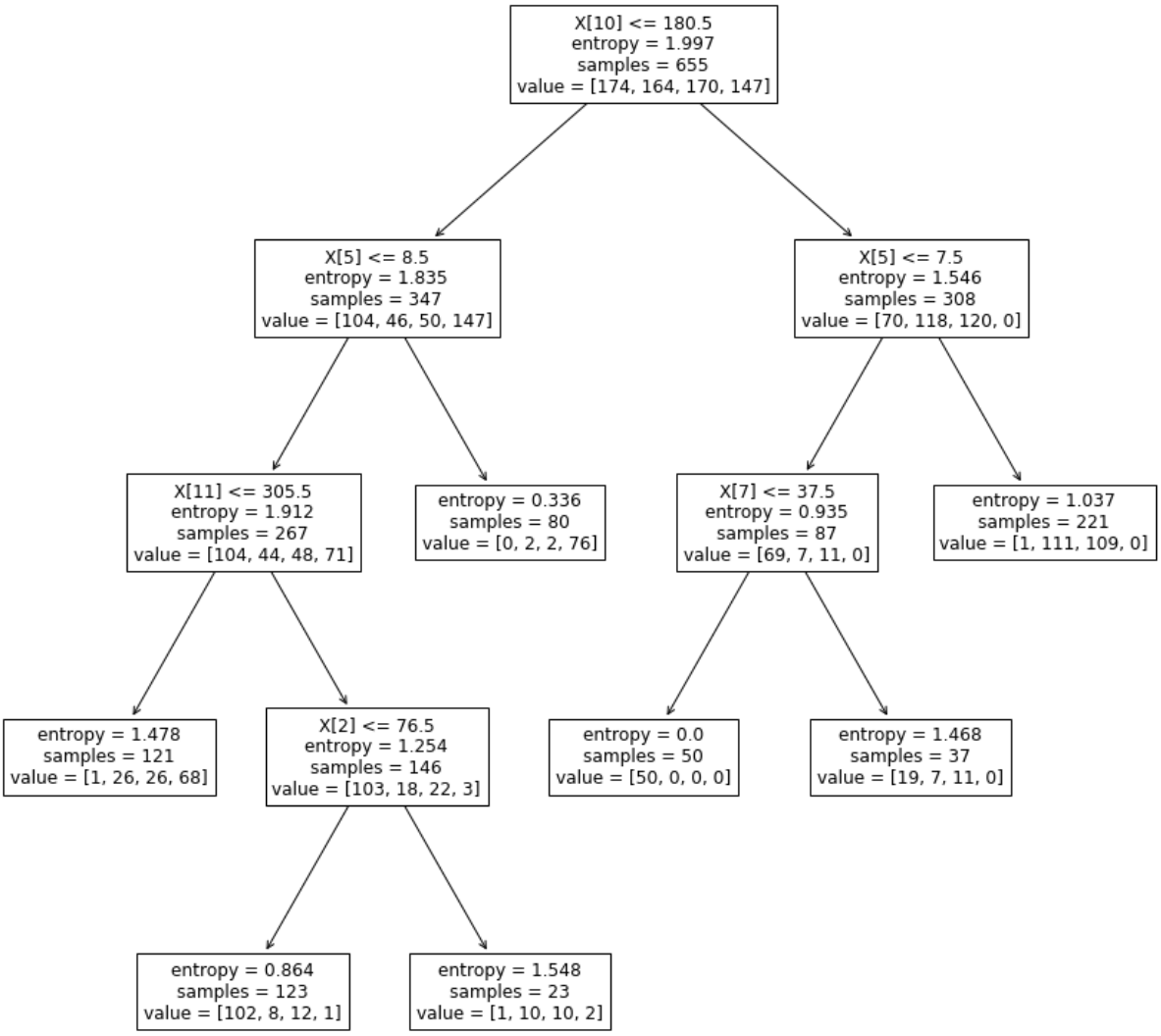
e) با استفاده از متد train_test_split و پارامتر random_state = 5 و با نسبت 0.8 به 0.2 داده های train و test هر یک از دیتافریم های x,y را جداسازی کرده و با نام های x_test, x_train, y_test, y_train برچسب گذاری می کنیم.

درصد توزیع هر یک از مقادیر فیلد Class را در دو بخش داده های یادگیری و تست بدست می آوریم :

```
==========> train <==========
Distribution Of saab : 25.95 %
Distribution Of bus  : 26.56 %
Distribution Of van  : 22.44 %
Distribution Of opel : 25.04 %
==========> test <==========
Distribution Of saab : 25.0 %
Distribution Of bus  : 18.9 %
Distribution Of van  : 27.44 %
Distribution Of opel : 28.66 %
```

همانطور که مشخص است درصد توزیع مقادیر گوناگون فیلد Class در هر دو داده های تست و یادگیری حدودا یکسان می باشد.

f) با استفاده از متد DecisionTreeClassifier و پارامتر های criterion = "entropy"و max_leaf_nodes = 7 درخت تصمیم را ایجاد کرده نمایش می دهیم. درخت تصمیم ایجاد شده به صورت زیر می باشد :




g) میزان دقت مدل بدست آمده حدود 0.64 (۶۴ درصد) می باشد.

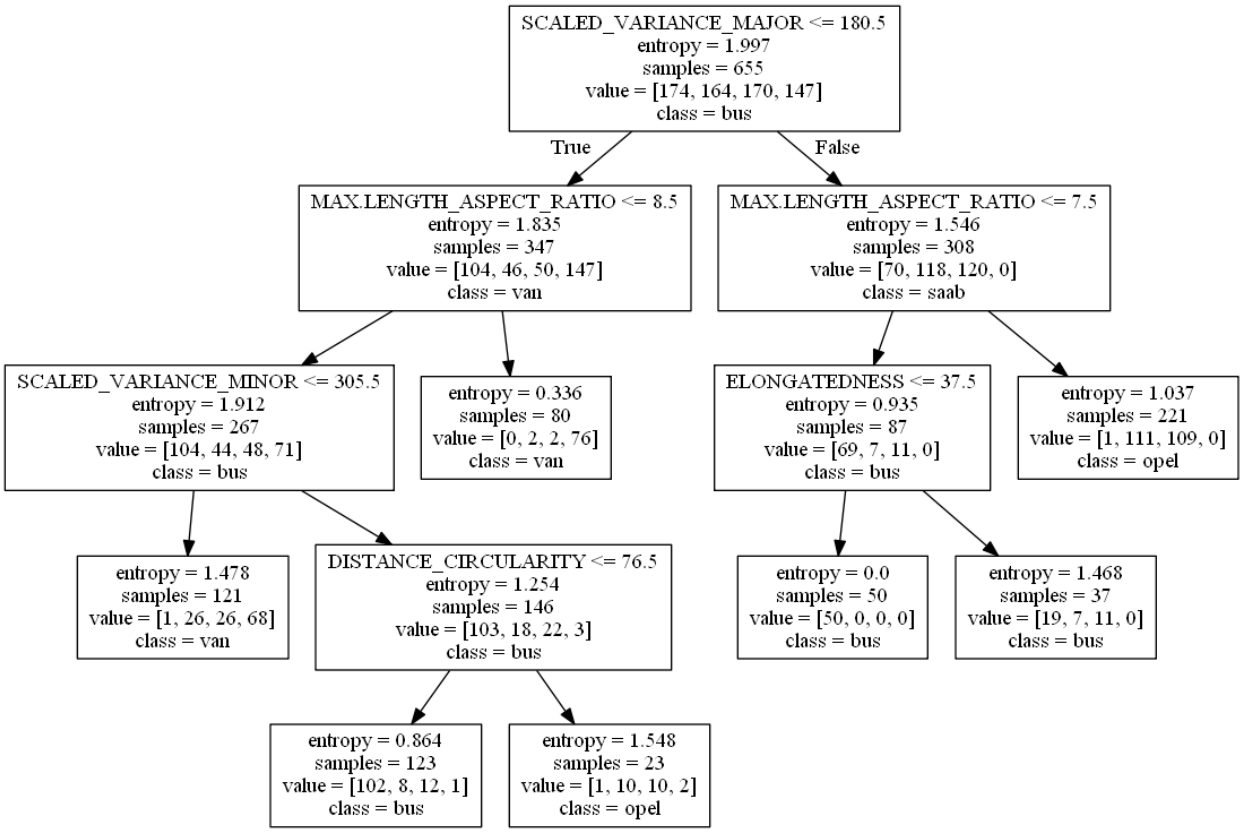
h) پارامتر های max_leaf_nodes, max_features در کلاس DecisionTreeClassifier:

max_leaf_nodes : حداکثر تعداد گره های برگ می باشد با توجه به این موضوع می توان گفت تا یک میزانی هرچه این پارامتر بیشتر باشد گره های برگ بدست آمده تعداد بیشتری داشته و لذا هر گره می تواند خالص تر باشد.

max_features : حداکثر تعداد فیلد ها و feature هایی که در ضمن split و جداسازی داده ها برای ساخت درخت مورد استفاده قرار می گیرد. تا حدودی می توان گفت هر چه میزان این پارامتر بزرگتر باشد و به نوعی تمام feature ها در جداسازی و ساخت درخت نقش داشته باشند مدل بدست آمده می تواند عملکرد بهتری داشته باشد.

گرچه افزایش هر دو پارامتر فوق می تواند منجر به کاهش سرعت تصمیم گیری مدل می شود چرا که در طول تست گره های بیشتر و feature های بیشتری باید تست و بررسی شود تا مدل نهایتا بتواند گره صحیح را برای یک رکورد بدست اورد.

i) خروجی تابع export_graphviz بر روی مدل ایجاد شده به صورت زیر می باشد :




تفسیر درخت : در ریشه متغیر scaled_variance_major با آستانه 180.5 منجر به جداسازی داده ها شده و درصورتی که داده تست دارای میزان scaled_variance_major کمتر از 180.5 باشد در گام بعدی متغیر max_length_aspect_ratio با آستانه 8.5 به عنوان جدا کننده ظاهر می شود و در صورتی که scaled_variance_major بزرگتر و مساوی 180.5 باشد این بار نیز متغیر max_length_aspect_ratio با آستانه 7.5 به عنوان جدا کننده ظاهر می گردد. به همین ترتیب در عمق های بعدی در زیر درخت سمت راست متغیر elongatedness و در زیر درخت سمت چپ متغیر scaled_variance_minor به عنوان جدا کننده ظاهر می شود. در ادامه زیر درخت سمت راست به گره های برگ رسیده و در داده های زیر درخت سمت چپ آن دسته از داده ها که میزان scaled_variance_major کمتر از 180.5 و scaled_variance_minor کمتر از 305.5 داشته اند به گره برگ رسیده و آن هایی کهscaled_variance_major کمتر از 180.5 و scaled_variance_minor بزرگتر

یا مساوی 305.5 داشته اند با استفاده از بررسی متغیر distance_circularity با آستانه 76.5 مجددا جدا سازی شده و به گره های برگ می رسند.

سوال ۲ )

a) با استفاده از متد read_csv و پارامتر ورودی names می توان برچسب ستون ها را تنظیم کرد.

b) ابتدا تمام missing value ها جستجو و با nan جایگذاری شده است. (البته به نظر دیتاست مربوطه فاقد missing value بوده است) سپس با استفاده از روش z_score داده های پرت هر ستون را حذف کرده و به این سبب که رنج مقادیر داده های ستون های مختلف دیتاست بسیار متفاوت است عمل استاندارد سازی داده ها را با استفاده از z_score انجام می دهیم. (برای مثال رنج اعداد ستون Diabetes pedigree function در بازه 0.078 تا 2.42 می باشد در حالی که رنج اعداد ستون Plasma glucose concentration a 2 hours in an oral glucose tolerance test در بازه 0 تا 199 می باشد) هیچ متغیر دسته ای در داخل دیتاست موجود نمی باشد لذا نیازی به encode کردن نبوده و نهایتا از دیتافریم ایجاد شده برای انجام پردازش ها در قسمت بعدی استفاده می نماییم.

(c

```
# 2-c

y = df_zscore[['Class variable (0 or 1)']]
x = df_zscore.drop(['Class variable (0 or 1)'], axis=1)
```

d) با استفاده از متد train_test_split و پارامتر random_state = 5 و با نسبت 0.8 به 0.2 داده های train و test هر یک از دیتافریم های x,y را جداسازی کرده و با نام های x_test, x_train, y_test, y_train برچسب گذاری می کنیم.

درصد توزیع هر یک از مقادیر فیلد Class را در دو بخش داده های یادگیری و تست بدست می آوریم :

```
==========> train <==========
Distribution Of 0 : 66.97 %
Distribution Of 1 : 33.03 %
==========> test <==========
Distribution Of 0 : 67.15 %
Distribution Of 1 : 32.85 %
```

همانطور که مشخص است درصد توزیع مقادیر گوناگون فیلد Class در هر دو داده های تست و یادگیری حدودا یکسان می باشد.

e) با استفاده از متد RandomForestClassifierو پارامتر ها n_estimators = 100 و criterion = "entropy"

و max_depth = 3 درخت تصمیم را ایجاد می کنیم.

f) میزان دقت مدل بدست آمده حدود 0.78 می باشد.

g) به طور کلی می توان گفت افزایش max_depth و حداکثر عمق درخت تصمیم می تواند باعث شود دسته بندی و جداسازی داده ها تا حد عمیق تری صورت گیرد و نهایتا گره های با خلوص بیشتری بدست آید. پس در حالت کلی تا یک آستانه ای افزایش max_depth می تواند افزایش دقت مدل را به همراه داشته باشد. همانطور که در تصویر زیر مشخص است اگر از max_depth = 1 آغاز کنیم تا رسیدن به max_depth = 7 افزایش حداکثر عمق به طور واضح منجر به افزایش میزان دقت از حدود 0.67 تا حدود 0.8 شده است اما از max_depth = 7 به بعد تقریبا میزان دقت ثابت شده و یا تغییرات اندکی داشته و بهترین دقت بدست امده با max_depth = 13 بوده است.

```
accuracy for max_depth  1 : 0.6715328467153284
accuracy for max_depth  2 : 0.7445255474452555
accuracy for max_depth  3 : 0.781021897810219
accuracy for max_depth  4 : 0.781021897810219
accuracy for max_depth  5 : 0.7956204379562044
accuracy for max_depth  6 : 0.781021897810219
accuracy for max_depth  7 : 0.8029197080291971
accuracy for max_depth  8 : 0.7956204379562044
accuracy for max_depth  9 : 0.7737226277372263
accuracy for max_depth  10 : 0.8029197080291971
accuracy for max_depth  11 : 0.8029197080291971
accuracy for max_depth  12 : 0.7956204379562044
accuracy for max_depth  13 : 0.8102189781021898
accuracy for max_depth  14 : 0.7956204379562044
```

سوال ۳ )

a) ابتدا تمام missing value ها جستجو و با nan جایگذاری شده است. (البته به نظر دیتاست مربوطه فاقد missing value بوده است) سپس با استفاده از روش z_score داده های پرت هر ستون را حذف کرده و به این سبب که رنج مقادیر داده های ستون های مختلف دیتاست بسیار متفاوت است عمل استاندارد سازی داده ها را با استفاده از z_score انجام می دهیم. (برای مثال رنج اعداد ستون Diabetes pedigree function در بازه 0.078 تا 2.42 می باشد در حالی که رنج اعداد ستون Plasma glucose concentration a 2 hours in an oral glucose tolerance test در بازه 0 تا 199 می باشد) هیچ متغیر دسته ای در داخل دیتاست موجود نمی باشد لذا نیازی به encode کردن نبوده و نهایتا از دیتافریم ایجاد شده برای تقسیم بندی داده ها به دو دسته test و train استفاده کرده و RandomForestClassifier را با پارامتر های n_estimators = 100, criterion = entropy و حداکثر عمق 13 که عمق درخت با بالاترین دقت بدست امده در سوال ۲ بود ایجاد می کنیم.

b) داده های تست را به مدل داده و نتایج بدست آمده برای ستون **Class variable (0 or 1)** را در متغیر y_pred ذخیره سازی میکنیم.

```
1  # 3_b
2  y_pred = rf01.predict(x_test)
```

c) نتیجه ماتریس confusion بدست آمده مطابق زیر است :

این ماتریس حاوی تعداد نتایج true positive, true negative, false positive, false negative بدست آمده از مقایسه نتایج موجود در y_pred با نتایج واقعی موجود در y_test می باشد.

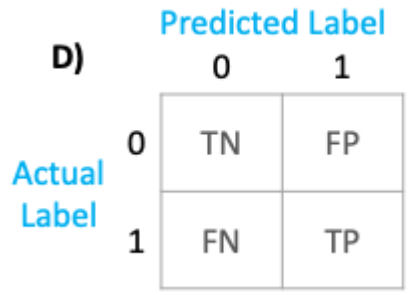
```
=====> model result without removing outliers <=====

array([[76, 16],
       [13, 32]], dtype=int64)
```

برای اینکه به شکل واضح این امر نشان داده شود با استفاده از یک حلقه می توانیم با صورت دستی تعداد TN , TP , FP , FN ها را محاسبه کنیم :

```
1  TN = 0; TP = 0; FN = 0; FP = 0
2  d = y_test.to_numpy()
3  row_num = y_test.shape[0]
4
5  for i in range(0, row_num):
6      if (d[i][0] == y_pred[i]) and (y_pred[i] == 0):
7          TN += 1
8      elif (d[i][0] == y_pred[i]) and (y_pred[i] == 1):
9          TP += 1
10     elif (d[i][0] != y_pred[i]) and (y_pred[i] == 0):
11         FN += 1
12     elif (d[i][0] != y_pred[i]) and (y_pred[i] == 1):
13         FP += 1
14
15 print("TN =",TN, "\tTP =", TP,"\tFN =", FN,"\tFP =", FP)
```

```
TN = 76         TP = 32         FN = 13         FP = 16
```

همانطور که از نتایج بدست آمده مشخص است ماتریس confusion به صورت زیر می باشد :




(d

Precision – What percent of your predictions were correct?

Recall – What percent of the positive cases did you catch?

F1 score – What percent of positive predictions were correct?

Support _ actual number of each value

```
print(classification_report(y_test, y_pred))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.85 | 0.83 | 0.84 | 92 |
| 1 | 0.67 | 0.71 | 0.69 | 45 |
| accuracy | | | 0.79 | 137 |
| macro avg | 0.76 | 0.77 | 0.76 | 137 |
| weighted avg | 0.79 | 0.79 | 0.79 | 137 |

همانطور که از نتایج به دست آمده مشخص است تعداد رکورد ها با ستون class variable = 0 در داخل داده های تست TN + FP = 92 مورد و تعداد رکورد ها با ستون class variable = 1 در داخل داده های تست تعداد TP + FN = 45 مورد بوده است.

Precision of 0 = TN / TPN = 76 / 89 = 0.85

Precision of 1 = TP / TPP = 32 / 48 = 0.66

Recall of 0 = TN / TAN = 76 / 92 = 0.83

Recall of 1 = TP / TAP = 32 / 45 = 0.71

F1-score of 0 = (2 * Precision of 0 * Recall of 0) / (Precision of 0 + Recall of 0)

= (2*0.85*0.83) / (0.85+0.83) = 1.411/ 1.67 = 0.84

F1-score of 1 = (2 * Precision of 1 * Recall of 1) / (Precision of 1 + Recall of 1)

= (2*0.67*0.71) / (0.67+0.71) = 0.9514 / 1.38 = 0.69

سوال ۴ )

X = (fever = yes, cough = no, headache = yes)

$$P(\text{cold} = \text{yes} \mid X) = \frac{P(\text{cold}=\text{yes} \cdot X)}{P(X)} = \frac{P(\text{cold} = \text{yes}) * P(X \mid \text{cold} = \text{yes})}{P(X)}$$

$$P(X \mid \text{cold} = \text{yes}) = P(\text{fever} = \text{yes} \mid \text{cold} = \text{yes}) * P(\text{cough} = \text{no} \mid \text{cold} = \text{yes}) * P(\text{headache} = \text{yes} \mid \text{cold} = \text{yes})$$

$$P(\text{fever} = \text{yes} \mid \text{cold} = \text{yes}) = \frac{3}{5}$$

$$P(\text{cough} = \text{no} \mid \text{cold} = \text{yes}) = \frac{1}{5}$$

$$P(\text{headache} = \text{yes} \mid \text{cold} = \text{yes}) = \frac{2}{5}$$

$$P(\text{cold} = \text{yes}) = \frac{5}{10}$$

$$P(\text{cold} = \text{yes} \mid X) = \frac{\frac{5}{10} * \frac{2}{5} * \frac{1}{5} * \frac{3}{5}}{P(X)} = \frac{\frac{3}{125}}{P(X)}$$

$$P(\text{cold} = \text{no} \mid X) = \frac{P(\text{cold}=\text{no} \,.\, X)}{P(X)} = \frac{P(\text{cold} = \text{no}) * P(X \mid \text{cold} = \text{no})}{P(X)}$$

$$P(X \mid \text{cold} = \text{no}) = P(\text{fever} = \text{yes} \mid \text{cold} = \text{no}) * P(\text{cough} = \text{no} \mid \text{cold} = \text{no}) * P(\text{headache} = \text{yes} \mid \text{cold} = \text{no})$$

$$P(\text{fever} = \text{yes} \mid \text{cold} = \text{no}) = \frac{2}{5}$$

$$P(\text{cough} = \text{no} \mid \text{cold} = \text{no}) = \frac{3}{5}$$

$$P(\text{headache} = \text{yes} \mid \text{cold} = \text{no}) = \frac{3}{5}$$

$$P(\text{cold} = \text{no}) = \frac{5}{10}$$

$$P(\text{cold} = \text{no} \mid X) = \frac{\frac{5}{10} * \frac{3}{5} * \frac{2}{5} * \frac{3}{5}}{P(X)} = \frac{\frac{9}{125}}{P(X)}$$

$$P(\text{cold} = \text{no} \,.\, X) > P(\text{cold} = \text{yes} \,.\, X) \rightarrow P(\text{cold} = \text{no} \mid X) > P(\text{cold} = \text{yes} \mid X)$$

با توجه به اینکه $P(\text{cold} = \text{no} \mid X)$ مقدار بزرگتری بدست آمده است لذا کسی دارای تب و عدم سرفه و دارای سردرد است پیش بینی می شود سرماخوردگی ندارد.