«*In The Name Of GOD*»

دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

[HW-03-Report]

[NEURAL COMPUTING AND DEEP LEARNING]

Hasan Masroor | [403131030] | April 26, 2025

"فهرست مطالب تمرين 03"

Question 1 …… 2

1) …….… 2

2) …….…… 12

3) ……... 15

4) …….….. 21

5) ……... 22

6) …….….. 23

Problem 3: Self Organizing Map (SOM)

Question 1:

.1

شبکه خودسازمانده کوهونن (SOM) یک روش یادگیری بدون نظارت است که برای کاهش ابعاد و خوشه‌بندی داده‌ها استفاده می‌شود و با حفظ ساختار داده‌ها، آن‌ها را به فضای کم‌بعدی منتقل می‌کند. در این تمرین هدف اصلی بررسی عملکرد SOM روی داده‌های بانکی است و ما باید ابتدا این شبکه را پیاده‌سازی کنیم و سپس با کاهش ابعاد داده به ۸، ۴ و ۲ بعد، عملکرد آن را با یک دسته‌بند ارزیابی نمایید. همچنین باید تأثیر پارامترهای مختلف مانند اندازه شبکه، نرخ یادگیری و ... را تحلیل کرده و نتایج را با روش‌هایی مثل PCA یا t-SNE مقایسه کنید؛ همچنین از SOM برای خوشه‌بندی داده‌ها بدون برچسب استفاده کرده و کیفیت خوشه‌بندی را با معیارهای استاندارد بسنجیم.

ابتدا کتابخانه‌های مورد نیاز را import کردیم و سپس دیتاست گفته شده را از UCI دریافت کردیم و فایل زیپ را استخراج نمودیم، سپس فایل CSV اصلی را با استفاده از ";" خواندیم و برای نمونه پنج نمونه اول این دیتاست را در خروجی نمایش دادیم:

```
   age        job  marital    education  default housing loan    contact  \
0   56  housemaid  married     basic.4y       no      no   no  telephone
1   57   services  married  high.school  unknown      no   no  telephone
2   37   services  married  high.school       no     yes   no  telephone
3   40     admin.  married     basic.6y       no      no   no  telephone
4   56   services  married  high.school       no      no  yes  telephone

  month day_of_week  ...  campaign  pdays  previous     poutcome emp.var.rate  \
0   may         mon  ...         1    999         0  nonexistent          1.1
1   may         mon  ...         1    999         0  nonexistent          1.1
2   may         mon  ...         1    999         0  nonexistent          1.1
3   may         mon  ...         1    999         0  nonexistent          1.1
4   may         mon  ...         1    999         0  nonexistent          1.1

   cons.price.idx  cons.conf.idx  euribor3m  nr.employed   y
0          93.994          -36.4      4.857       5191.0  no
1          93.994          -36.4      4.857       5191.0  no
2          93.994          -36.4      4.857       5191.0  no
3          93.994          -36.4      4.857       5191.0  no
4          93.994          -36.4      4.857       5191.0  no

[5 rows x 21 columns]
```

همان‌طور که در تصویر بالا هم مشاهده می‌کنیم برخی ستون‌ها شامل مقادیر unknown هستند و همچنین ترکیبی از ستون‌های عددی و غیرعددی داریم و قبل از پاسخ به سوال اول لازم است که پیش پردازش‌های مورد نیاز را انجام دهیم. اگر اندازه دیتاست رو نمایش دهیم می‌بینیم که 21 ستون و 41188 ردیف داریم:

```
(41188, 21)
```

در ادامه مقادیر unknown در هر ستون از داده‌های بانکی را شمارش کردیم و فقط ستون‌هایی که دارای این مقادیر بودند را نمایش دادیم تا دید کلی از داده‌های گم‌شده در مجموعه داده به دست آوریم:

```
Unknown values in each column:
job           330
marital        80
education    1731
default      8597
housing       990
loan          990
dtype: int64
```

وقتی به تصویر بالا دقت کنیم می‌بینیم که شش ستون حاوی مقادیر unknown هستند و حال برای هندل کردن این مقادیر، برای هر ستون مقدار unknown را با مُد (پرتکرارترین مقدار) همان ستون جایگزین نمودیم تا داده‌های گم‌شده را به شکلی معنادار پر کنیم و بعدا برای حل پارت های مختلف تمرین و مدل som مشکلی ایجاد نشود. در نهایت نیز پنج ردیف اول داده‌های اصلاح‌شده و تعداد باقیمانده مقادیر 'unknown' را نمایش دادیم تا از صحت عملیات جایگزینی اطمینان حاصل کنیم.

در خروجی زیر می‌بینیم که دیگر ستون‌های حاوی unknown نداریم و تا حد خوبی توانستیم این مشکل مقادیر گمشده را حل کنیم:

```
Unknown values in each column:
Series([], dtype: int64)
```

سپس در ادامه در خروجی زیر 5 ردیف اول پس از اصلاح مشکل بالا را نمایش می‌دهیم:

```
   age        job  marital    education default housing loan    contact month  \
0   56  housemaid  married     basic.4y      no      no   no  telephone   may
1   57   services  married  high.school      no      no   no  telephone   may
2   37   services  married  high.school      no     yes   no  telephone   may
3   40     admin.  married     basic.6y      no      no   no  telephone   may
4   56   services  married  high.school      no      no  yes  telephone   may

  day_of_week  ...  campaign  pdays  previous     poutcome emp.var.rate  \
0         mon  ...         1    999         0  nonexistent         1.1
1         mon  ...         1    999         0  nonexistent         1.1
2         mon  ...         1    999         0  nonexistent         1.1
3         mon  ...         1    999         0  nonexistent         1.1
4         mon  ...         1    999         0  nonexistent         1.1

   cons.price.idx  cons.conf.idx  euribor3m  nr.employed   y
0          93.994          -36.4      4.857       5191.0  no
1          93.994          -36.4      4.857       5191.0  no
2          93.994          -36.4      4.857       5191.0  no
3          93.994          -36.4      4.857       5191.0  no
4          93.994          -36.4      4.857       5191.0  no

[5 rows x 21 columns]
```

در این مرحله، ستون‌های غیرعددی موجود در دیتاست را شناسایی کردیم تا برای پیش‌پردازش‌های بعدی آماده شوند و این ستون‌ها به صورت زیر هستند:

```
Non-numeric columns: Index(['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact',
       'month', 'day_of_week', 'poutcome', 'y'],
      dtype='object')
```

در ادامه برای حل این مورد نیز ابتدا ستون‌های باینری مثل 'default' و'y' را با استفاده از LabelEncoder به مقادیر عددی 0 و 1 تبدیل کردیم و سپس برای ستون‌هایی که بیشتر از دو مقدار دارند مثل 'job' و'education' از کدگذاری One-Hot استفاده کردیم که هر مقدار منحصر به فرد را به یک ستون مجزا با مقادیر 0 یا 1 تبدیل می‌کند. در نهایت نیز پنج ردیف اول را برای بررسی بهتر در خروجی زیر نمایش می‌دهیم:

```
   age  default  housing  loan  duration  campaign  pdays  previous  \
0   56        0        0     0       261         1    999         0
1   57        0        0     0       149         1    999         0
2   37        0        1     0       226         1    999         0
3   40        0        0     0       151         1    999         0
4   56        0        0     1       307         1    999         0

   emp.var.rate  cons.price.idx  ...  month_oct  month_sep  day_of_week_mon  \
0           1.1          93.994  ...      False      False             True
1           1.1          93.994  ...      False      False             True
2           1.1          93.994  ...      False      False             True
3           1.1          93.994  ...      False      False             True
4           1.1          93.994  ...      False      False             True

   day_of_week_thu  day_of_week_tue  day_of_week_wed  poutcome_nonexistent  \
0            False            False            False                  True
1            False            False            False                  True
2            False            False            False                  True
3            False            False            False                  True
4            False            False            False                  True

   poutcome_success  marital_married  marital_single
0             False             True           False
1             False             True           False
2             False             True           False
3             False             True           False
4             False             True           False

[5 rows x 48 columns]
```

اگر به شکل بالا دقت کنیم می‌بینیم که تعداد ستون‌های ما افزایش پیدا کرده و از 21 به 48 ستون رسیده است.
در مرحله بعد با استفاده از روش نرمال‌سازی Min-Max تمام مقادیر را در بازه 0 تا 1 قرار دادیم تا مقیاس ویژگی‌ها یکسان شود و در ادامه بتوانیم تحلیل‌های دقیق‌تر و بهتری داشته باشیم و مجدد نمونه‌ای از داده‌ها را در خروجی زیر نمایش دادیم:

```
        age  default  housing  loan  duration  campaign  pdays  previous  \
0  0.481481      0.0      0.0   0.0  0.053070       0.0    1.0       0.0
1  0.493827      0.0      0.0   0.0  0.030297       0.0    1.0       0.0
2  0.246914      0.0      1.0   0.0  0.045954       0.0    1.0       0.0
3  0.283951      0.0      0.0   0.0  0.030704       0.0    1.0       0.0
4  0.481481      0.0      0.0   1.0  0.062424       0.0    1.0       0.0

   emp.var.rate  cons.price.idx  ...  month_oct  month_sep  day_of_week_mon  \
0        0.9375        0.698753  ...        0.0        0.0              1.0
1        0.9375        0.698753  ...        0.0        0.0              1.0
2        0.9375        0.698753  ...        0.0        0.0              1.0
3        0.9375        0.698753  ...        0.0        0.0              1.0
4        0.9375        0.698753  ...        0.0        0.0              1.0

   day_of_week_thu  day_of_week_tue  day_of_week_wed  poutcome_nonexistent  \
0              0.0              0.0              0.0                   1.0
1              0.0              0.0              0.0                   1.0
2              0.0              0.0              0.0                   1.0
3              0.0              0.0              0.0                   1.0
4              0.0              0.0              0.0                   1.0

   poutcome_success  marital_married  marital_single
0               0.0              1.0             0.0
1               0.0              1.0             0.0
...
3               0.0              1.0             0.0
4               0.0              1.0             0.0

[5 rows x 48 columns]
```

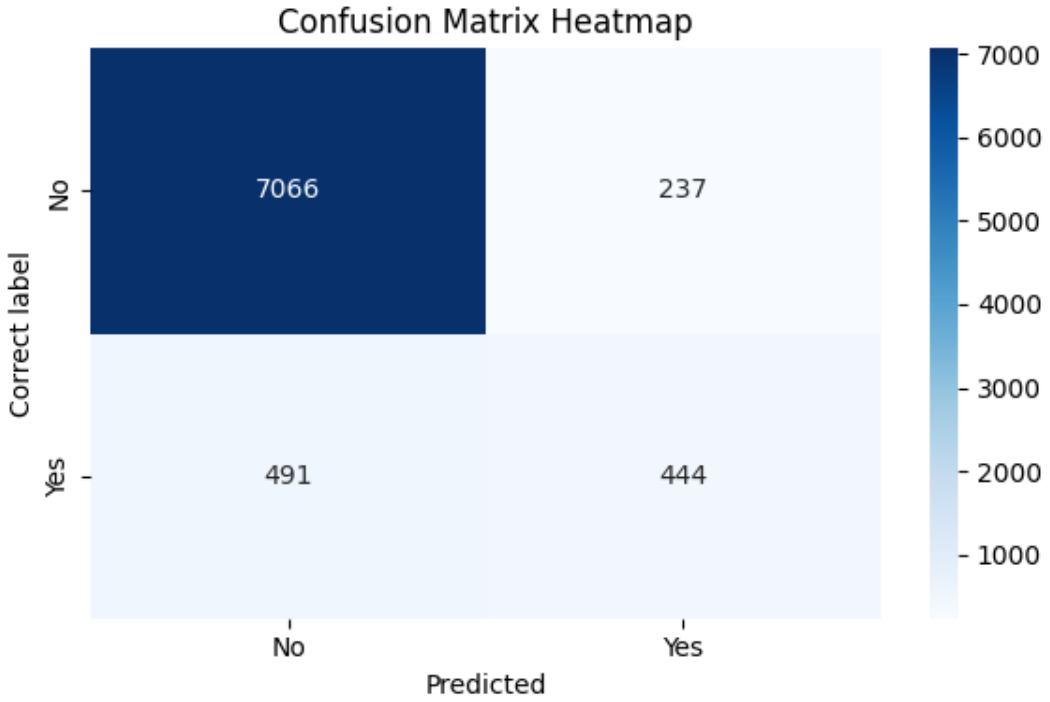
در نهایت هم دیتاست پیش پردازش شده را به دو مجموعه تست و آموزش که به ترتیب 20 درصد و 80 درصد از دیتاست را شامل می‌شوند تقسیم کردیم:

```
Train shape: (32950, 47)
Test shape: (8238, 47)
```

پس از اینکه پیش پردازش‌های مورد نیاز و همچنین تقسیم داده‌ها را انجام دادیم حال کلاس SOM را پیاده‌سازی کردیم که یک شبکه خودسازمانده کوهونن را شبیه‌سازی می‌کند. ابتدا در تابع **init** پارامترهای اصلی مدل شامل ابعاد شبکه، بعد ورودی، نرخ یادگیری، شعاع همسایگی و تعداد تکرارها را مقداردهی تعریف کردیم؛ سپس با استفاده از تابع find_bmu ، بهترین واحد تطابق (BMU) را برای هر داده ورودی پیدا می‌کنیم که نزدیک‌ترین نورون به آن داده است. در تابع neighborhood_func، میزان تأثیر همسایگی‌ها بر اساس فاصله از BMU و

تکرار فعلی محاسبه می‌شود. در مرحله آموزش نیز وزن‌های شبکه به‌صورت تدریجی بر اساس داده‌های ورودی و تابع همسایگی به‌روز می‌شوند و در نهایت توابع transform و visualize به ترتیب برای تبدیل داده‌ها به مختصات BMU و نمایش بصری وزن‌های شبکه استفاده می‌شوند. این پیاده‌سازی امکان کاهش ابعاد و خوشه‌بندی داده‌ها را فراهم می‌کند.

پس از اینکه مدل som را تعریف کردیم قبل از اینکه کاهش بعد به 8، 4 و 2 را انجام دهیم طبق گفته سوال اول آموزش بدون کاهش بعد را انجام می‌دهیم و پس از نمایش نتایج و تجزیه و تحلیل به کاهش بعد با مقادیر گفته شده خواهیم پرداخت. در این مرحله ابتدا مدل Random Forest را آموزش دادیم و زمان اجرای آن را محاسبه کردیم. سپس با استفاده از داده‌های آزمون، پیش‌بینی‌های مدل را انجام دادیم و معیارهای ارزیابی مختلف شامل دقت (Accuracy) ، Adjusted Rand Index (ARI)، و Silhouette Score را محاسبه نمودیم و در نهایت نیز ماتریس درهم ریختگی (Confusion Matrix) را رسم کردیم تا عملکرد مدل در پیش‌بینی کلاس‌های مختلف را به‌صورت بصری نمایش دهیم. این معیارها به ما کمک می‌کنند تا کیفیت مدل را از جنبه‌های مختلف ارزیابی کنیم. همان‌طور که ماتریس زیر می‌بینیم 7066تا از کلاس NO را به درستی تشخیص داده است و همچنین 444تا از کلاس‌های Yes را درست تشخیص داده است و مثل فرمول‌های تمرین‌های گذشته به کمک مقادیر ماتریس می‌توانیم مقدار دقت و ... را نیز بدست آوریم:

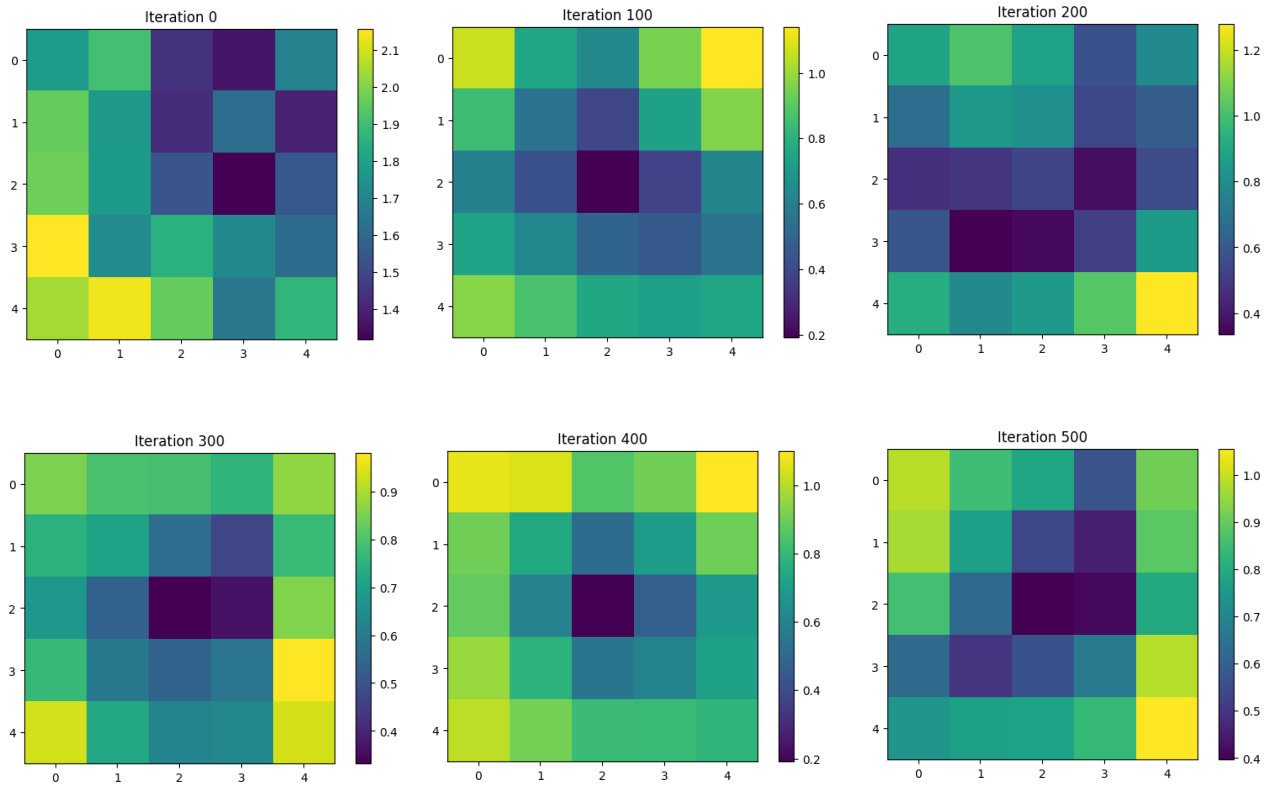


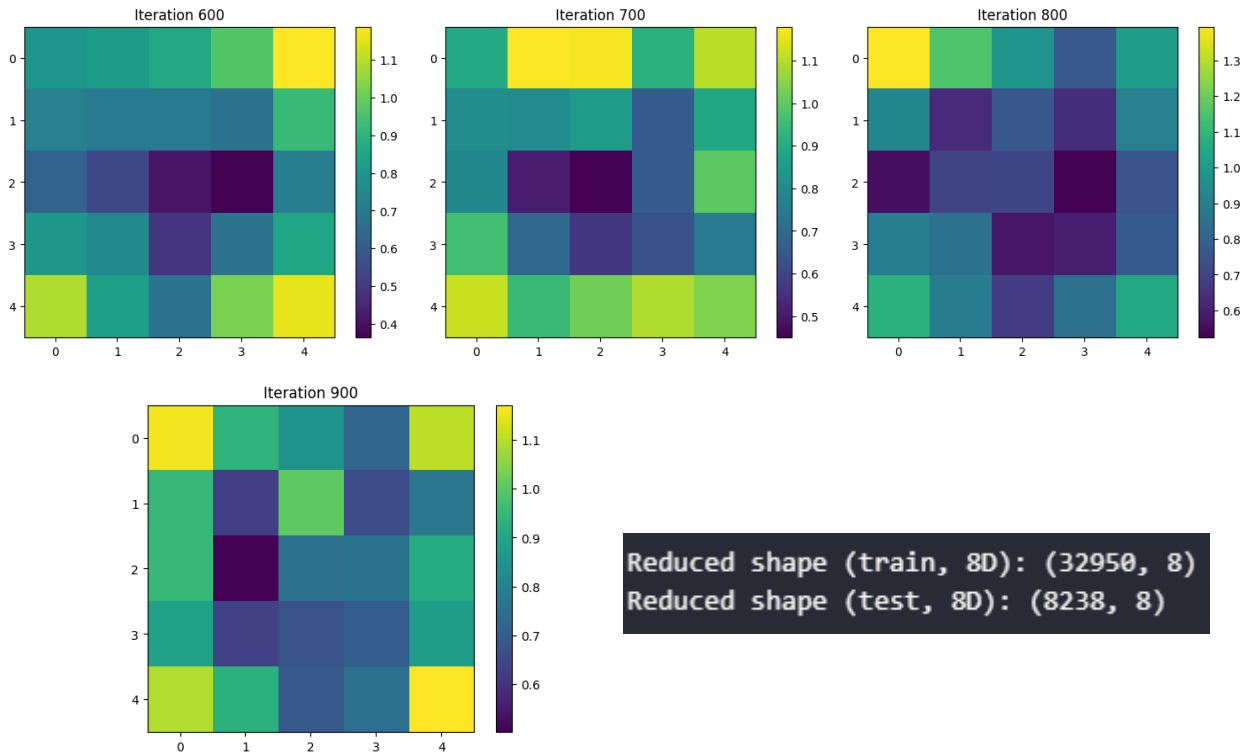
در شکل پایین نیز سایر معیارها را داریم و می‌بینیم این مدل در مدت زمان نسبتاً کوتاه آموزش دیده و به دقت قابل قبول 91.016% پیش‌بینی نتایج دست یافته است که نشان‌دهنده عملکرد خوب مدل در طبقه‌بندی داده‌هاست. مقدارARI حاکی از آن است که مدل تا حدی توانسته ساختار خوشه‌های واقعی داده را شناسایی کند و از طرفی Silhouette Score پایین نشان می‌دهد که خوشه‌بندی انجام‌شده چندان متمایز نیست و احتمالاً همپوشانی

قابل‌توجهی بین خوشه‌ها وجود دارد. به‌طور کلی می‌توانیم بگوییم مدل در طبقه‌بندی عملکرد خوبی داشته ولی در خوشه‌بندی داده‌ها چندان متمایز عمل نکرده است:

```
Training Time: 5.5987 seconds
Accuracy: 0.9116
Adjusted Rand Index (ARI): 0.4479
Silhouette Score: 0.0756
```

در ادامه بر می‌گردیم به کاهش بعد با استفاده از som و در این بخش، ابتدا یک شبکه خودسازمانده (SOM) با ابعاد 5×5 نورون آموزش دادیم تا الگوهای موجود در داده‌ها را شناسایی کند. سپس با محاسبه فاصله هر داده تا نزدیک‌ترین نورون‌های شبکه، داده‌های اصلی را به بردارهای 8 بعدی جدید تبدیل کردیم که نمایانگر موقعیت داده‌ها در فضای نقشه SOM هستند. این تبدیل باعث کاهش ابعاد داده‌ها شد، در حالی که اطلاعات مکانی و ساختاری آن‌ها حفظ گردید. در نهایت نیز خروجی این بخش را در خروجی نمایش می‌دهیم:





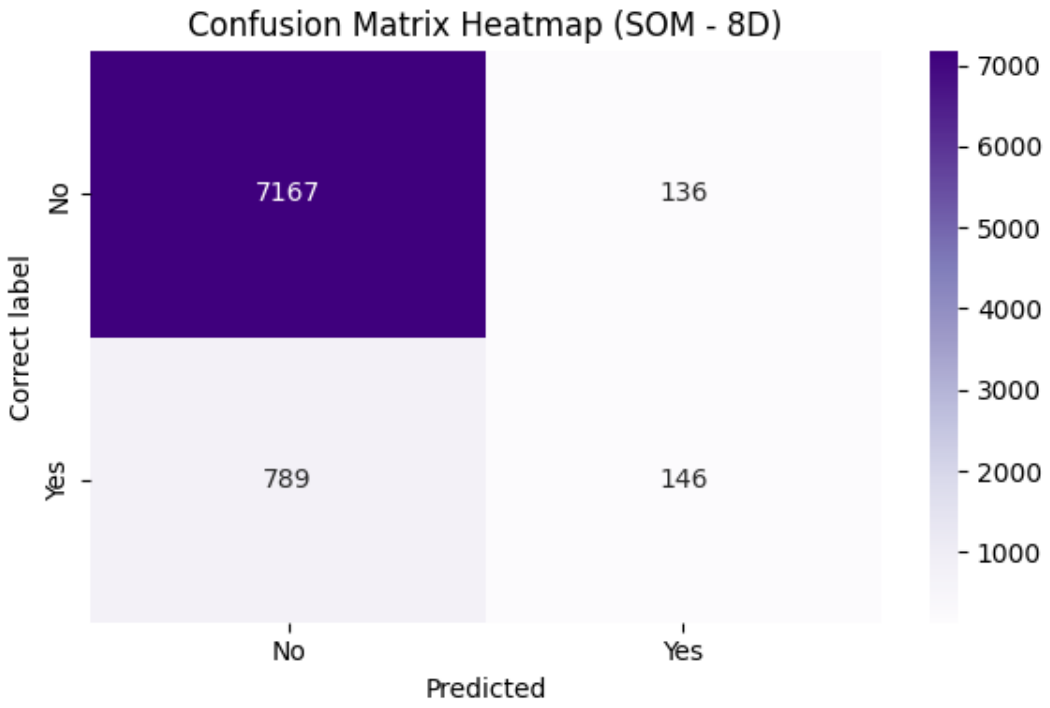

```
Reduced shape (train, 8D): (32950, 8)
Reduced shape (test, 8D): (8238, 8)
```

در ادامه یک مدل Random Forest روی داده‌های کاهش‌یافته ۸ بعدی آموزش دادیم و زمان اجرای آن را ثبت کردیم. سپس با استفاده از داده‌های آزمون، پیش‌بینی‌های مدل را انجام دادیم و معیارهای ارزیابی مختلف را محاسبه نمودیم و در نهایت نیز ماتریس درهم ریختگی را رسم کردیم تا عملکرد مدل در پیش‌بینی کلاس‌های مختلف را به‌صورت بصری نمایش دهیم.

در شکل زیر مشاهده می‌کنیم که پس از کاهش ابعاد به ۸ بعد باSOM ، در مدت زمان حدودا 19 ثانیه آموزش دیده و به دقت 88.77% در پیش‌بینی نتایج دست یافته است که نشان‌دهنده عملکرد نسبتاً خوب مدل در طبقه‌بندی داده‌هاست، هرچند نسبت به حالت قبل کاهش دقت مشاهده می‌شود. مقدار ARI نیز حاکی از همبستگی نسبتاً قوی بین خوشه‌بندی انجام‌شده و ساختار واقعی داده‌هاست. از سوی دیگر، Silhouette Score نیز نشان‌دهنده بهبود قابل‌توجهی در تمایز خوشه‌ها نسبت به مرحله قبل است و اگرچه هنوز فضای برای بهبود وجود دارد. به‌طور کلی می‌توان گفت کاهش ابعاد با SOM منجر به حفظ ساختار داده‌ها شده و مدل توانسته هم در طبقه‌بندی و هم در خوشه‌بندی عملکرد مناسبی ارائه دهد، هرچند این کاهش ابعاد باعث افت جزئی در دقت طبقه‌بندی شده است:

```
[SOM with 8 dimensions]
Training Time: 19.1805 seconds
Accuracy: 0.8877
Adjusted Rand Index (ARI): 0.8877
Silhouette Score: 0.5223243381340082
```

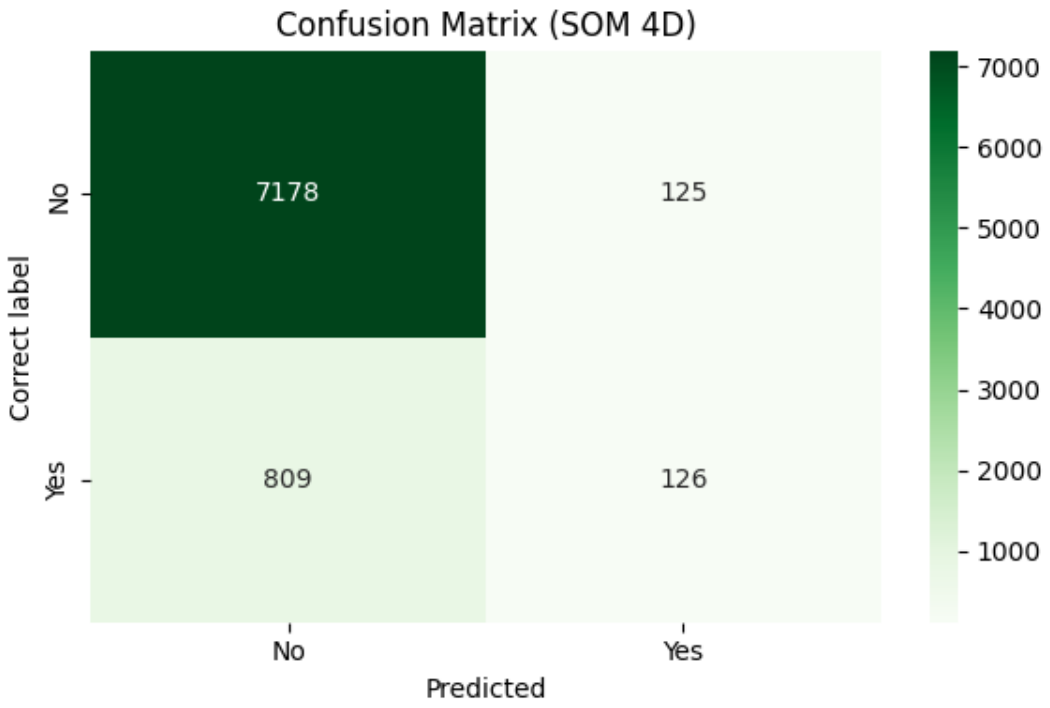
در ماتریس زیر می‌بینیم 7167تا از کلاس NO را به درستی تشخیص داده است و همچنین 146تا از کلاس‌های Yes را درست تشخیص داده است و مثل فرمول‌های تمرین‌های گذشته به راحتی می‌توانیم که به کمک مقادیر ماتریس می‌توانیم مقدار دقت و ... را نیز بدست آوریم:



در ادامه، همان مراحل قبلی را این بار برای کاهش ابعاد به 4 بعد تکرار کردیم. ابتدا داده‌ها را با SOM به فضای 4 بعدی تبدیل نمودیم، سپس یک مدل Random Forest روی این داده‌های کاهش‌یافته آموزش دادیم و در نهایت عملکرد مدل را با معیارهای مختلف ارزیابی و نتایج را تحلیل کردیم. همان‌طور که در خروجی زیر می‌بینیم، نتایج نشان می‌دهند مدل پس از کاهش ابعاد به 4 بعد، در مدت زمان حدودا 21 ثانیه آموزش دیده و به دقت 88.66% دست یافته که عملکردی تقریبا مشابه حالت 8 بعدی دارد. مقدار ARI نیز همبستگی ضعیف‌تر بین خوشه‌بندی و ساختار واقعی داده نسبت به مرحله قبل را نشان می‌دهد، در حالی که مقدار Silhouette Score حاکی از کیفیت نسبتاً مناسب اما نه چندان مطلوب در تمایز خوشه‌هاست. به طور کلی، کاهش بیشتر ابعاد اگرچه تأثیر محسوسی بر دقت طبقه‌بندی نداشته اما باعث کاهش کیفیت خوشه‌بندی شده است:

```
[SOM with 4 dimensions]
Training Time: 21.6311 s
Accuracy: 0.8866
ARI:      0.1500
Silhouette: 0.4918652146844992
```

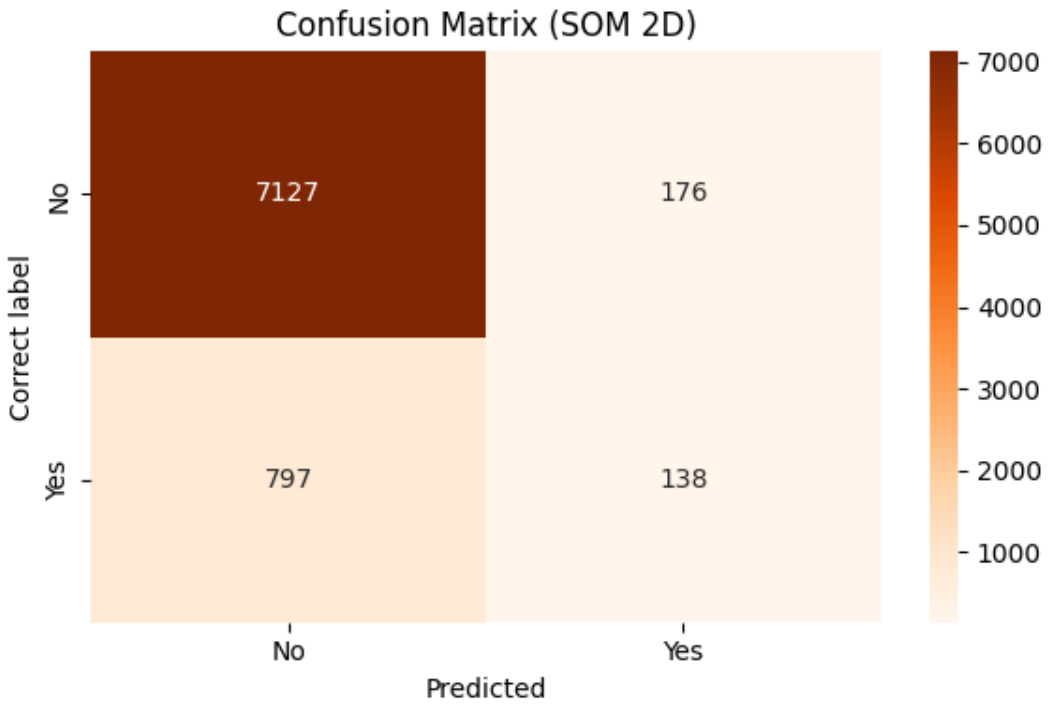
ماتریس زیر را نیز مثل موارد قبلی مجدد می‌توانیم تجزیه و تحلیل کنیم و می‌بینیم که 7178تا از کلاس NO را به درستی تشخیص داده است و همچنین 126تا از کلاس‌های Yes را درست تشخیص داده است و مثل فرمول‌های تمرین‌های گذشته به راحتی می‌توانیم که به کمک مقادیر ماتریس می‌توانیم مقدار دقت و ... را نیز بدست آوریم:



در ادامه، همان مراحل قبلی را این بار برای کاهش ابعاد به 4 بعد تکرار کردیم. ابتدا داده‌ها را با SOM به فضای 4 بعدی تبدیل نمودیم، سپس یک مدل Random Forest روی این داده‌های کاهش‌یافته آموزش دادیم و در نهایت عملکرد مدل را با معیارهای مختلف ارزیابی و نتایج را تحلیل کردیم. شکل خروجی زیر نیز نتایج کاهش ابعاد به 2 بعد نشان می‌دهد و مدل در زمان کوتاه‌تر و در حدود 13 ثانیه آموزش دیده و به دقت 88.19% رسیده که کاهش جزئی نسبت به مراحل قبل دارد. مقدار ARI نیز نشان می‌دهد مدل همچنان تا حدی توانسته ساختار خوشه‌های واقعی را شناسایی کند. مقدار Silhouette Score که نسبت به مراحل قبل کاهش یافته، حاکی از کاهش کیفیت و تمایز خوشه‌ها در فضای 2 بعدی است. این نتایج نشان می‌دهد هرچه ابعاد را بیشتر کاهش دهیم، اگرچه زمان آموزش کمتر می‌شود اما ممکن است به کیفیت خوشه‌بندی لطمه بخورد، در حالی که دقت طبقه‌بندی تغییر محسوسی نمی‌کند. به طور کلی مدل در فضای 2 بعدی هم عملکرد قابل قبولی در طبقه‌بندی دارد اما برای کاربردهای حساس به خوشه‌بندی ممکن است نیاز به ابعاد بالاتری داشته باشیم:

```
[SOM with 2 dimensions]
Training Time: 13.5843 s
Accuracy: 0.8819
ARI: 0.1499
Silhouette: 0.44640699260924427
```

ماتریس زیر را نیز مثل موارد قبلی مجدد می‌توانیم تجزیه و تحلیل کنیم و می‌بینیم که 7127تا از کلاس NO را به درستی تشخیص داده است و همچنین 138تا از کلاس‌های Yes را درست تشخیص داده است و مثل فرمول‌های تمرین‌های گذشته به راحتی می‌توانیم که به کمک مقادیر ماتریس می‌توانیم مقدار دقت و ... را نیز بدست آوریم:

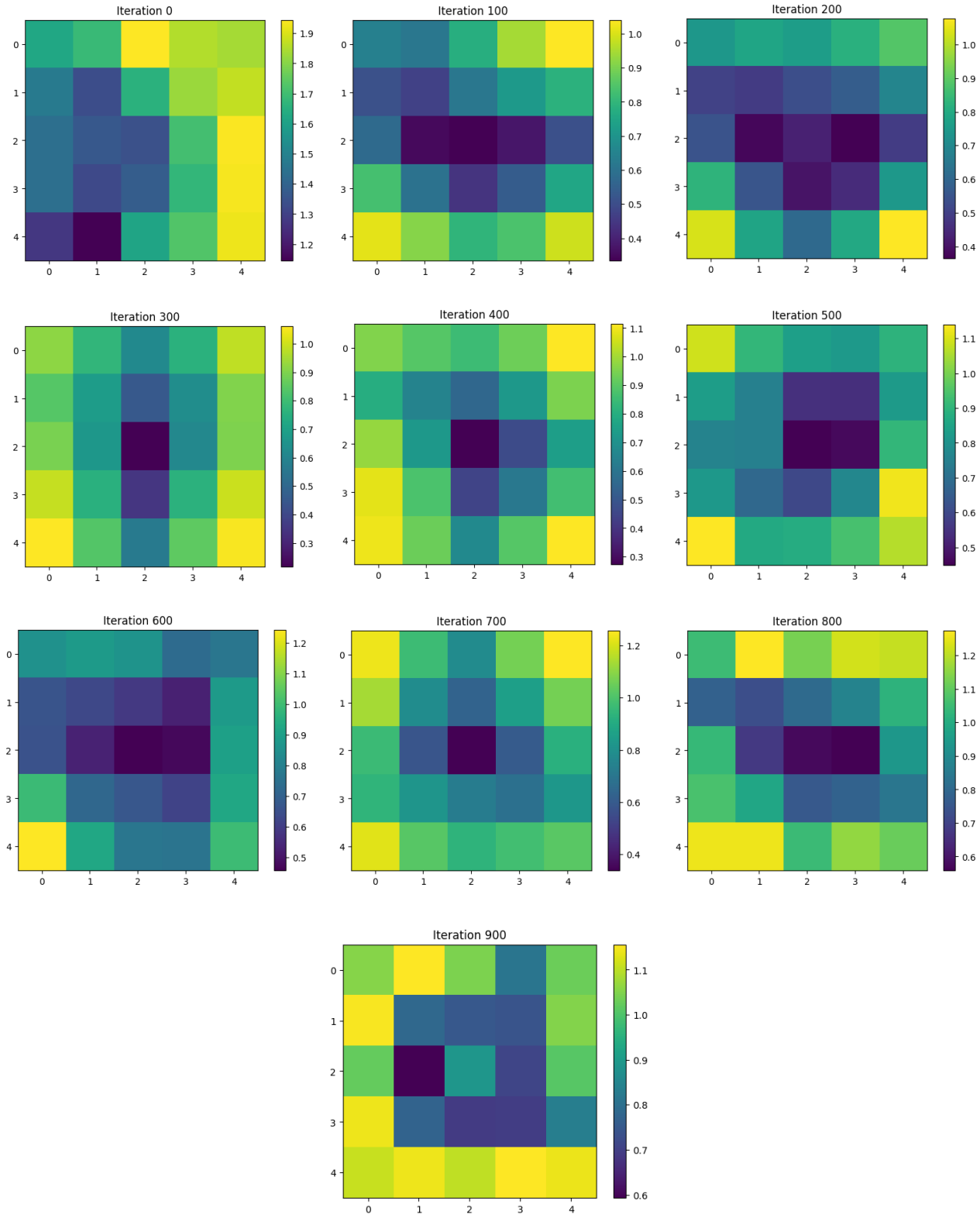


از تحلیل‌های بالا به این نتیجه رسیدیم که آموزش دسته‌بند بدون کاهش بعد دقت حدود 91% درصد داشت و با کاهش بعد به 8 دقت به حدود 88% رسید و با ادامه کاهش بعد به 4 و 2 این دقت به صورت جزئی کاهش داشت و همینطور زمان آموزش مدل هم با کاهش بعد کمتر شد و هرچه به سمت بعد 2 پیش رفتیم مدل به مدت زمان کمتری برای آموزش نیاز پیدا کرد اما از آن طرف کیفیت خوشه بندی کاهش پیدا کرد و طبق توضیحاتی که قسمت‌های قبلی دادیم یک تریدآفی داریم و در کل کاهش بعد به ابعاد گفته شده مدل عملکرد خوبی از خود نشان داد و دقت هر سه مدل حدود 88% شد.

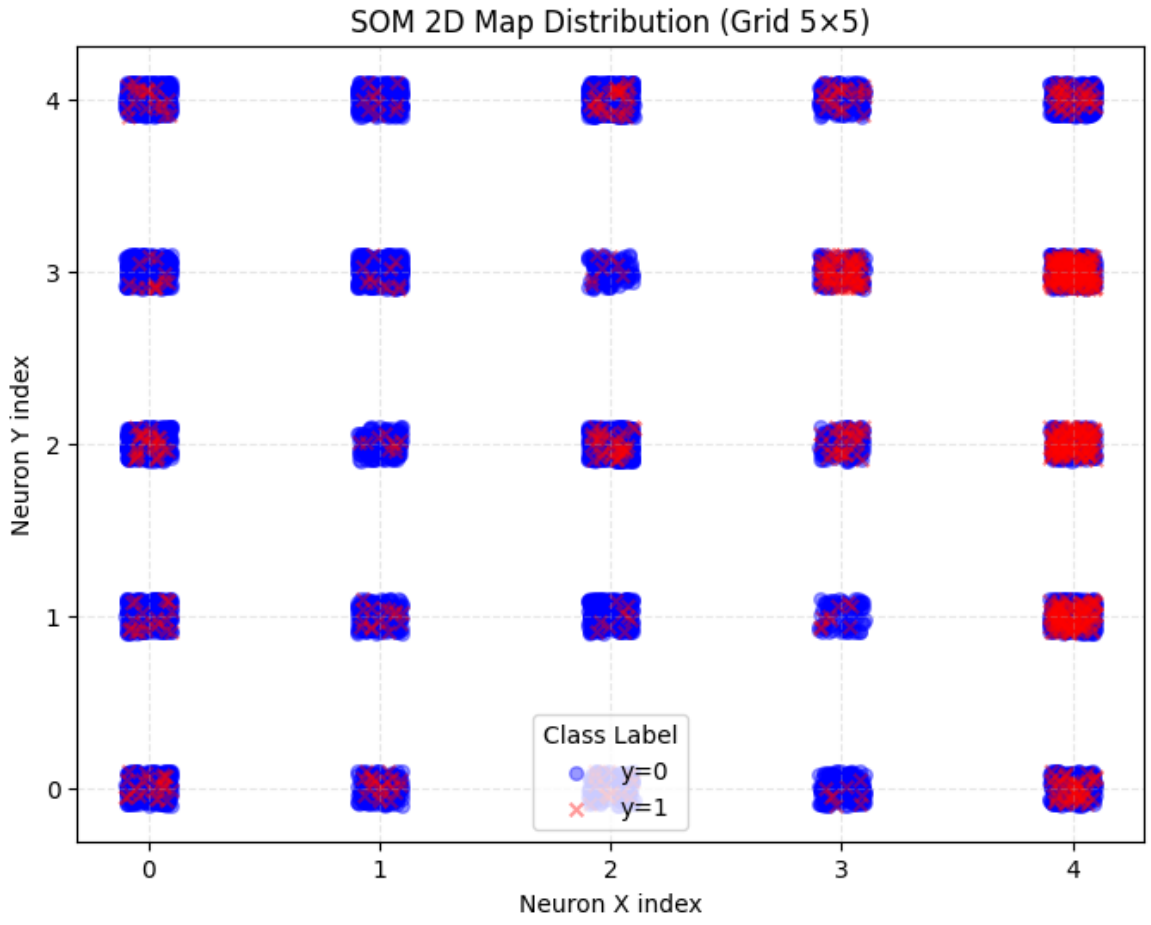
.2

در این پارت باید با استفاده از شبکه SOM یک نقشه 2 بعدی از توزیع داده‌ها ایجاد می‌کنیم تا الگوهای پنهان و ساختار کلی داده‌ها را بصری‌سازی کنیم. ابتدا شبکه SOM با ابعاد 5×5 نورون آموزش دادیم تا ساختار داده‌ها را یاد بگیرد. سپس برای هر نمونه از داده‌های آزمون، بهترین نورون تطبیق‌یافته (BMU) را در نقشه SOM پیدا کردیم و مختصات آن‌ها را استخراج نمودیم. برای بهبود نمایش بصری، مقدار کمی نویز تصادفی به مختصات

اضافه کردیم تا از همپوشانی نقاط جلوگیری شود و در نهایت نیز با رسم نقاط بر اساس لیبل‌های واقعی توزیع داده‌ها را روی نقشه SOM بصری‌سازی کردیم و به ازای هر 100 تکرار در خروجی داریم:



در این نقشه‌ی ۲ بعدی SOM با ساختار 5×5، داده‌های مربوط به دو کلاس y=0 و y=1 پس از کاهش بعد به صورت خوشه‌هایی پیرامون نورون‌های شبکه پخش شده‌اند. هر نورون در این شبکه نقش یک مرکز تجمع برای داده‌هایی با ویژگی‌های مشابه را دارد. مشاهده می‌شود که نمونه‌های هر دو کلاس در اکثر نورون‌ها حضور دارند اما در برخی نقاط تراکم یکی از کلاس‌ها بیشتر است که بیانگر تمایل داده‌های آن کلاس به ویژگی‌های خاص آن نورون می‌باشد (در بعضی نورون‌ها دایره‌های آبی که نشانگر کلاس 0 هستند زیاد هستند و در بعضی دیگر از نورون‌ها که در شکل زیر هم می‌بینیم ضربدرهای قرمز که همان کلاس 1 هستند بیشتر دیده می‌شوند). این توزیع نشان می‌دهد که ویژگی‌های دو کلاس در بعضی ابعاد همپوشانی دارند اما در برخی نواحی، SOM موفق شده آن‌ها را از هم تفکیک کند. در مجموع این شبکه خودسازمانده ما توانسته ساختار درونی داده‌ها و شباهت‌های نسبی بین کلاس‌ها را در قالب یک نمای فشرده و قابل تحلیل حفظ کند و تحلیل‌های انجام شده در بالا را می‌توانیم در شکل زیر نیز به طور کامل مشاهده کنیم:



.3

در این پارت باید تأثیر پارامترهای کلیدی SOM شامل اندازه شبکه، نرخ یادگیری و تعداد تکرارهای آموزش را بر کیفیت کاهش ابعاد بررسی کنیم. برای این کار، مقادیر مختلفی برای هر پارامتر تست شده و نتایج حاصل از نظر معیارهای ارزیابی (مانند دقت طبقه‌بندی و کیفیت خوشه‌بندی) با استفاده از نمودارها یا جداول مقایسه‌ای تحلیل و ارائه می‌شود. ابتدا سایز شبکه را 5x5 در نظر گرفتیم و پارامترهای دیگر را تغییر دادیم و تاثیر را در ادامه برررسی می‌کنیم. در تست اول مقدار نرخ یادگیری 0.5 و تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 5x5]
Accuracy: 0.8858
ARI:      0.1787
Silhouette: 0.5420
Training Time: 18.1099 seconds
```

در تست دوم مقدار نرخ یادگیری را تغییر دادیم و برابر 0.1 قرار دادیم و با تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 5x5]
Accuracy: 0.8877
ARI:      0.1847
Silhouette: 0.5014
Training Time: 19.7897 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 500 کاهش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 5x5]
Accuracy: 0.8869
ARI:      0.1565
Silhouette: 0.3407
Training Time: 21.1132 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 2000 افزایش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 5x5]
Accuracy: 0.8864
ARI:      0.1684
Silhouette: 0.5009
Training Time: 23.6451 seconds
```

سپس آمدیم برای تجزیه و تحلیل بهتر خروجی‌های بالا که برای یک شبکه 5x5 بود را در جدول زیر نمایش دادیم و تحلیل نتایج نشان‌دهنده تأثیر متفاوت تنظیم پارامترهای SOM بر عملکرد مدل است. بهترین ترکیب پارامتری در این آزمایش‌ها، نرخ یادگیری 0.1 با 1000 تکرار بوده که بالاترین مقدار ARI و دقت را به همراه Silhouette Score نسبتا خوب ارائه داده است. کاهش تعداد تکرارها به 500 موجب افت محسوس کیفیت خوشه‌بندی شده و دقت را هم جزئی کاهش داده است در حالی که افزایش آن به 2000 بهبود قابل‌توجهی ایجاد نکرده و تنها زمان آموزش را افزایش داده است. نرخ یادگیری 0.5 نیز نتایج نسبتاً مطلوبی داشته اما نسخه 0.1 آن عملکرد کمی بهتر نشان داده است (در صورت افزایش نرخ یادگیری تا 1 احتمال کاهش دقت و ... به خاطر نرخ بالا و آموزش نامطلوب بیشتر می‌شود). جالب توجه اینکه تغییرات پارامترها تأثیر محدودی بر دقت نهایی مدل داشته و این مقدار در تمام آزمایش‌ها حول 88% نوسان داشته است که نشان‌دهنده پایداری نسبی مدل در برابر تغییر پارامترهاست. موارد گفته شده را در جدول زیر می‌توانیم ببینیم:

اندازه شبکه	5x5	5x5	5x5	5x5
نرخ یادگیری	0.5	0.1	0.5	0.5
تعداد تکرار	1000	1000	500	2000
ARI	%17.87	%18.47	%15.65	%16.84
Silhouette Score	%54.2	%50.14	%34.07	%50.09
Training Time	18.1 s	19.78 s	21.11 s	23.64 s
دقت نهایی	%88.58	%88.77	%88.69	%88.64

در ادامه سایز شبکه را 10x10 در نظر گرفتیم و پارامترهای دیگر را تغییر دادیم و تاثیر را در ادامه برررسی می‌کنیم.
در تست اول مقدار نرخ یادگیری 0.5 و تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 10x10]
Accuracy: 0.8874
ARI: 0.1615
Silhouette: 0.4136
Training Time: 22.0060 seconds
```

در تست دوم مقدار نرخ یادگیری را تغییر دادیم و برابر 0.1 قرار دادیم و با تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 10x10]
Accuracy: 0.8837
ARI: 0.1147
Silhouette: 0.3679
Training Time: 20.3869 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 500 کاهش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 10x10]
Accuracy: 0.8859
ARI: 0.1569
Silhouette: 0.4621
Training Time: 22.2125 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 2000 افزایش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 10x10]
Accuracy: 0.8883
ARI: 0.1790
Silhouette: 0.4761
Training Time: 20.7746 seconds
```

سپس آمدیم برای تجزیه و تحلیل بهتر خروجی‌های بالا که برای یک شبکه 10x10 بود را در جدول زیر نمایش دادیم و تحلیل نتایج نشان‌دهنده تأثیر متفاوت تنظیم پارامترهای SOM بر عملکرد مدل است. تحلیل نتایج جدول برای شبکه‌ی 10x10 نشان‌دهنده‌ی برخی تفاوت‌ها نسبت به شبکه‌ی 5x5 است. در این تنظیمات، افزایش اندازه‌ی شبکه به 10x10 موجب کاهش نسبی مقادیر ARI در مقایسه با شبکه‌ی کوچک‌تر شده است، به‌طوری‌که بهترین مقدار ARI با 2000 تکرار و نرخ یادگیری 0.5 به‌دست آمده که نسبت به حالت 5x5 بهبود جزئی نشان می‌دهد. نکته‌ی قابل‌توجه این‌است که Silhouette Score در این اندازه‌ی شبکه بهبود یافته و به 47.61% رسیده است که نشان‌گر تمایز خوب خوشه‌ها در فضای با ابعاد بزرگ‌تر است. با این‌حال، نرخ یادگیری 0.01 در این شبکه نتایج ضعیف‌تری را در مقایسه با شبکه‌ی کوچک‌تر نشان داده و ARI را کاهش داده است. در مقابل کاهش تعداد تکرارها به 500 تأثیر محسوسی بر کیفیت خوشه‌بندی نداشته است؛ زمان آموزش نیز در این شبکه تقریباً ثابت بوده که نشان‌دهنده‌ی کارایی مناسب مدل حتی در ابعاد بزرگ‌تر است. دقت نهایی مدل در تمام آزمایش‌ها حول 88% ثابت مانده که مجدداً تأیید می‌کند تغییر پارامترها تأثیر چندانی بر دقت طبقه‌بندی ندارد. به‌طور کلی، به‌نظر می‌رسد افزایش اندازه‌ی شبکه اگرچه می‌تواند کیفیت خوشه‌بندی را بهبود بخشد، اما لزوماً منجر به افزایش تطابق با برچسب‌های واقعی (ARI) نمی‌شود و انتخاب اندازه‌ی شبکه باید با توجه به هدف تحلیل (خوشه‌بندی یا طبقه‌بندی) صورت گیرد. موارد گفته شده را در جدول زیر می‌توانیم ببینیم:

اندازه شبکه	10x10	10x10	10x10	10x10
نرخ یادگیری	0.5	0.1	0.5	0.5
تعداد تکرار	1000	1000	500	2000
ARI	%16.15	%11.47	%15.69	%17.9
Silhouette Score	%41.36	%36.79	%46.21	%47.61
Training Time	22.00 s	20.38 s	22.21 s	20.77 s
دقت نهایی	%88.74	%88.37	%88.59	%88.83

در نهایت نیز سایز شبکه را 15x15 در نظر گرفتیم و پارامترهای دیگر را تغییر دادیم و تاثیر را در ادامه برررسی می‌کنیم. در تست اول مقدار نرخ یادگیری 0.5 و تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 15x15]
Accuracy: 0.8888
ARI:      0.1748
Silhouette: 0.4362
Training Time: 19.2706 seconds
```

در تست دوم مقدار نرخ یادگیری را تغییر دادیم و برابر 0.1 قرار دادیم و با تعداد تکرار 1000 را اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 15x15]
Accuracy: 0.8874
ARI:      0.1615
Silhouette: 0.4584
Training Time: 23.2587 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 500 کاهش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 15x15]
Accuracy: 0.8846
ARI: 0.0939
Silhouette: 0.3123
Training Time: 23.3029 seconds
```

در تست سوم مقدار نرخ یادگیری 0.5 و تعداد تکرار را به 2000 افزایش دادیم و اجرا کردیم که خروجی زیر حاصل شد:

```
[Grid 15x15]
Accuracy: 0.8853
ARI: 0.1388
Silhouette: 0.3432
Training Time: 17.8907 seconds
```

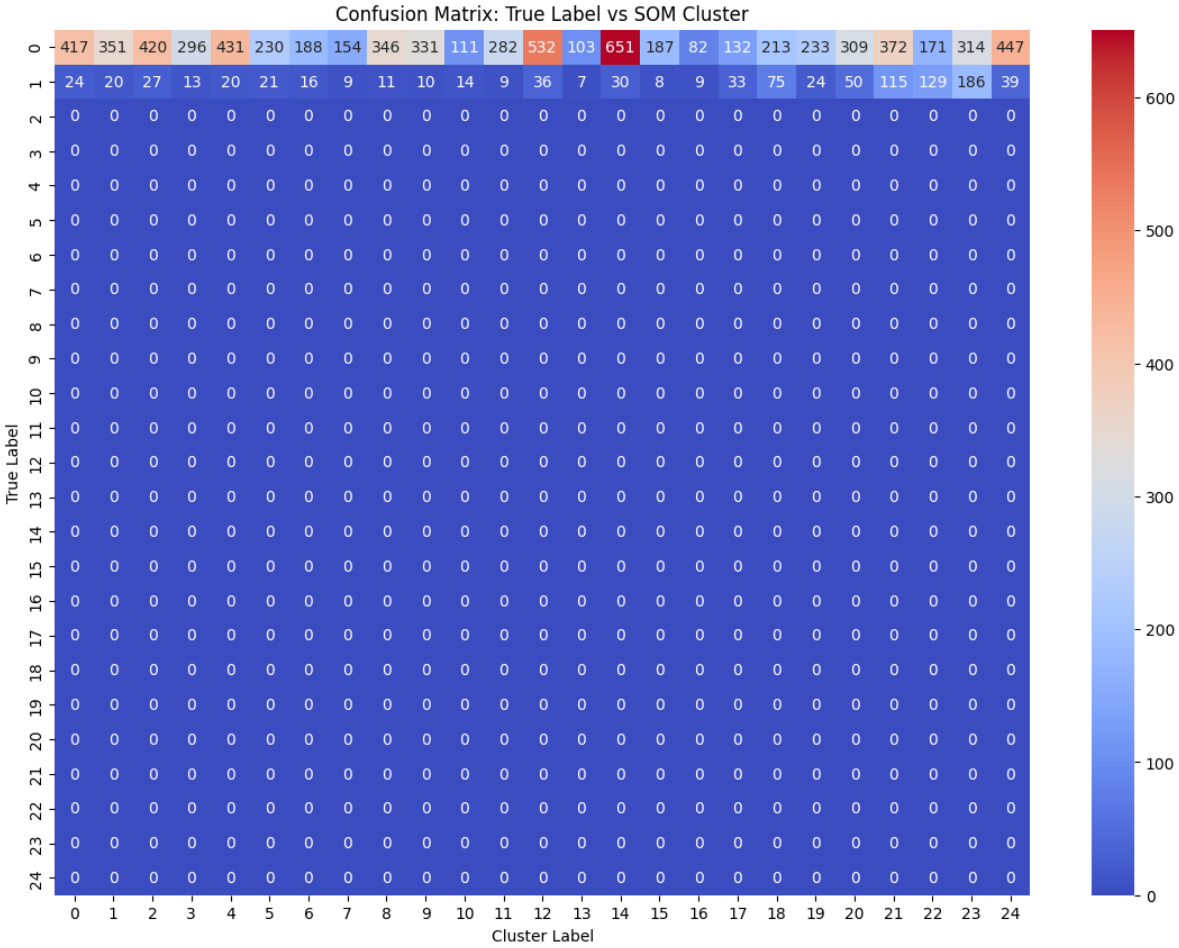
سپس آمدیم برای تجزیه و تحلیل بهتر خروجی‌های بالا که برای یک شبکه 10x10 بود را در جدول زیر نمایش دادیم و تحلیل نتایج نشان‌دهنده تأثیر متفاوت تنظیم پارامترهای SOM بر عملکرد مدل است.

تحلیل نتایج شبکه 15x15 نشان‌دهنده رفتار پیچیده‌تر و جالب‌تری نسبت به اندازه‌های کوچک‌تر است. در این ابعاد بزرگ‌تر شبکه، بهترین عملکرد از نظر ARI (17.48%) با نرخ یادگیری 0.5 و 1000 تکرار حاصل شده که نسبت به شبکه‌های 5x5 و 10x10 بهبود محسوسی را نشان می‌دهد؛ با این حال نتایج نشان می‌دهد افزایش تعداد تکرارها به 2000 برخلاف انتظار منجر به کاهش ARI به 13.88% شده که احتمالاً نشان‌دهنده پدیده بیش‌برازش در ابعاد بزرگ شبکه است. از سوی دیگر، کاهش تعداد تکرارها به 500 موجب افت شدید هر دو معیار خوشه‌بندی شده که حساسیت بالای این ابعاد شبکه به تعداد تکرارهای آموزش را آشکار می‌سازد، همچنین زمان آموزش در این شبکه بین 17 تا 23 ثانیه متغیر بوده است. همانند آزمایش‌های قبلی، دقت نهایی مدل حول 88% ثابت مانده که نشان‌دهنده استحکام و پایداری مدل در برابر تغییر ابعاد شبکه است. در مجموع، شبکه 15x15 در برخی تنظیمات خاص (نرخ یادگیری 0.5 با 1000 تکرار) بهترین مقادیر ARI را در بین تمام اندازه‌های آزمایش‌شده ارائه داده است، اما ناپایداری نتایج در سایر تنظیمات نشان می‌دهد که کار با این ابعاد بزرگ نیازمند تنظیم دقیق‌تر پارامترهاست. این مشاهدات بر اهمیت تعادل دقیق بین اندازه شبکه و پارامترهای آموزشی تأکید دارد و نشان می‌دهد که انتخاب بهینه این پارامترها باید با توجه به هدف نهایی تحلیل (خوشه‌بندی دقیق یا طبقه‌بندی پایدار) و با در نظر گرفتن منابع محاسباتی موجود باید انجام پذیرد. موارد گفته شده را در جدول زیر می‌توانیم ببینیم:

اندازه شبکه	15x15	15x15	15x15	15x15
نرخ یادگیری	0.5	0.1	0.5	0.5
تعداد تکرار	1000	1000	500	2000
ARI	%17.48	%16.15	%9.39	%13.88
Silhouette Score	%43.62	%45.84	%31.23	%34.32
Training Time	19.27 s	23.25 s	23.30 s	17.89 s
دقت نهایی	%88.88	%88.74	%88.46	%88.53

.4

در این بخش باید از شبکه خودسازمانده برای خوشه‌بندی داده‌ها بدون استفاده از برچسب‌ها استفاده کنیم، سپس خوشه‌های ایجادشده را با برچسب‌های واقعی مقایسه کرده و کیفیت خوشه‌بندی را با معیارهای ارزیابی تحلیل نماییم. ابتدا با استفاده از SOM داده‌های تست را به خوشه‌ها اختصاص دادیم. برای این کار، ابتدا بهترین نورون‌های تطبیق‌یافته (BMU) برای هر نمونه را یافته و سپس با ترکیب مختصات آن‌ها در شبکهSOM ، برچسب‌های خوشه را ایجاد کردیم. در مرحله بعد نیز کیفیت خوشه‌بندی را با معیارهای ARIو Silhouette Score بررسی کردیم. در نهایت نیز ماتریس درهم ریختگی را رسم کردیم که ارتباط بین برچسب‌های واقعی و برچسب‌های خوشه را به صورت بصری نمایش می‌دهد:



در این خروجی، شبکه‌ی SOM داده‌ها را به صورت بدون نظارت خوشه‌بندی کرده است و نتایج در قالب یک ماتریس درهم ریختگی نمایش داده شده‌اند. همان‌طور که مشاهده می‌شود تنها ردیف‌های ۰ و ۱ دارای مقادیر غیر صفر هستند، در حالی که سایر ردیف‌ها کاملاً صفر باقی مانده‌اند. این نشان می‌دهد که کل داده‌ها فقط شامل دو کلاس (افتتاح حساب یا عدم افتتاح حساب) بوده‌اند و هیچ داده‌ای با برچسب‌های دیگر در مجموعه وجود

نداشته است. به طور کلی این نتایج نشان می‌دهند که SOM توانسته ساختار پنهان داده‌ها را تا حد مناسبی یاد بگیرد اما همچنان مقداری همپوشانی و پراکندگی در خوشه‌بندی وجود دارد.

در شکل زیر نیز نتایج نشان می‌دهند که خوشه‌بندی انجام‌شده توسط SOM بدون استفاده از برچسب‌ها، ارتباط ضعیفی با ساختار واقعی داده‌ها دارد؛ همچنین مقدار پایین Silhouette Score حاکی از آن است که خوشه‌های ایجادشده فاقد ساختار درونی متمایز هستند و نقاط به خوبی از هم تفکیک نشده‌اند. این نتایج نشان می‌دهد که SOMدر این تنظیمات نتوانسته الگوی معناداری در داده‌ها شناسایی کند که ممکن است به دلیل پیچیدگی ذاتی توزیع داده‌ها باشد:

```
Unsupervised Clustering with SOM:
Adjusted Rand Index (ARI): 0.0033
Silhouette Score: 0.04069527781048046
```

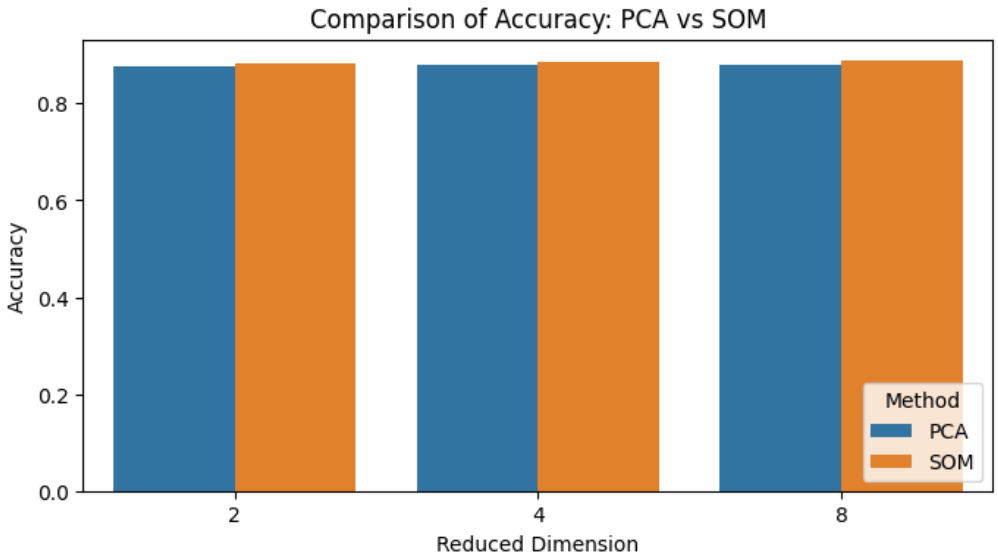
.5

در این مرحله باید عملکرد شبکه خودسازمانده را با روش‌های مرسوم کاهش ابعاد مثل PCA یا t-SNE مقایسه کنیم. در این بخش ابتدا یک فرآیند سیستماتیک برای مقایسه دو روش کاهش ابعاد PCA و SOM طراحی کردیم. برای این کار، داده‌های آموزشی و آزمون را به ابعاد مختلف کاهش دادیم و سپس یک مدل Random Forest روی داده‌های کاهش‌یافته آموزش دادیم. در ادامه، معیارهای ارزیابی مختلف که در بخش‌های قبل با آنها کار کردیم و آشناییت داریم را برای هر حالت محاسبه و ذخیره کردیم. در نهایت نیز نتایج حاصل را در خروجی زیر نمایش دادیم:

	method	dim	train_time	accuracy	ari	silhouette
0	PCA	8	21.210346	0.879340	0.194945	0.041123
1	PCA	4	22.911953	0.879097	0.190374	0.082125
2	PCA	2	12.311839	0.876426	0.162581	0.167150
3	SOM	8	18.128609	0.887715	0.171909	0.522324
4	SOM	4	25.755002	0.886623	0.150032	0.491865
5	SOM	2	13.279101	0.881889	0.149861	0.446407

نتایج مقایسه نشان می‌دهد که هر دو روش PCA و SOM عملکرد نسبتاً مشابهی در حفظ دقت طبقه‌بندی دارند، با این تفاوت که SOM در ابعاد بالاتر (8 بعدی) کمی دقت بهتری (88.77%) نسبت به PCA با (87.93%) ارائه می‌دهد. از نظر زمان آموزش، PCA در ابعاد پایین‌تر سریع‌تر عمل کرده، در حالی که SOM در ابعاد بالاتر زمان بیشتری نیاز دارد. نکته جالب توجه تفاوت معنادار در معیار Silhouette Score است که

نشان می‌دهد SOM توانسته ساختار خوشه‌ای بهتری ایجاد کند و این نشان می‌دهد SOM در حفظ ساختار محلی داده‌ها برتری دارد. از طرفی، ARI در PCA کمی بالاتر است که نشانگر تطابق بهتر با برچسب‌های واقعی است. به طور کلی هر دو روش عملکرد قابل قبولی دارند و اگر کیفیت خوشه‌بندی مهم باشد، SOM گزینه بهتری محسوب می‌شود. تحلیل‌های بالا را در قالب یک نمودار بر اسا دقت برای درک بهتر در زیر نمایش دادیم و می‌توانیم در سه بعد 8، 4 و 2 در این دو مدل دقت کسب شده را ببینیم:



6.

این پارت از ما می‌خواهد یک مدل چندلایه از شبکه‌های SOM طراحی کنیم که لایه‌های بالاتر آن بتوانند مناطق پرتراکم داده را با دقت بیشتری تحلیل کنند و ساختار داده‌ها را بهتر نمایش دهند. در این کد ابتدا یک SOM اولیه با شبکه 5x5 آموزش دادیم تا ساختار کلی داده‌ها را یاد بگیرد. سپس داده‌های آموزشی را بر اساس نورون‌های فعال در این شبکه دسته‌بندی کرده و برای هر ناحیه پرتراکم (با حداقل 5% داده)، یک SOM ثانویه 3x3 آموزش دادیم. در نهایت با تابع hierarchical_embedding ، هر نمونه داده را به یک بردار چهار بعدی تبدیل کردیم که شامل مختصات نورون فعال در SOM اولیه و در صورت وجود، مختصات مربوطه در SOM ثانویه است. این رویکرد سلسله‌مراتبی امکان تحلیل دقیق‌تر نواحی پرتراکم داده را فراهم می‌کند. در ادامه یک مدل Random Forest روی داده‌های تبدیل‌شده به فضای سلسله‌مراتبی آموزش دادیم و معیارهای دقت(Accuracy) ، تطابق خوشه‌ها (ARI) و کیفیت خوشه‌بندی (Silhouette) و همچنین زمان آموزش مدل را محاسبه کردیم تا عملکرد این روش را ارزیابی کنیم.

خروجی این بخش به صورت زیر در آمد و نتایج نشان می‌دهد مدل سلسله‌مراتبی با وجود مزایای نظری، در عمل عملکرد ضعیف‌تری نسبت به SOM ساده داشته است. دقت پایین‌تر احتمالاً به دلیل پیچیدگی بیش از حد مدل برای این مجموعه داده خاص است که منجر به کاهش توان تعمیم‌پذیری شده است. مقدار ARI نسبتاً پایین نیز نشان می‌دهد ارتباط ضعیفی بین خوشه‌های شناسایی‌شده Silhouette Score بسیار پایین نیز تأیید می‌کند که

خوشه‌ها به خوبی از هم تفکیک نشده‌اند. همچنین زمان آموزش کوتاه‌تر ممکن است به دلیل کاهش ابعاد موثر داده در روش سلسله‌مراتبی باشد. به نظر می‌رسد این معماری برای این مجموعه داده نیاز به تنظیم دقیق‌تر پارامترها دارد اما به دلیل به پایان رسیدن ددلاین این تمرین و نبودن فرصت کافی، فرصتی برای بهبود و افزایش این معیارهای ارزیابی در پارت ششم مهیا نشد و خروجی این بخش نیز به شرح زیر می‌باشد:

```
Hierarchical SOM Embedding:
Accuracy: 0.7645
ARI:      0.1592
Silhouette: 0.11374255096235171
Training Time: 2.9253 seconds
```

«. . . اردیبهشت‌ماه ۱۴۰۴ . . .»