Domain Adaptation using Unsupervised Learning methods

Computational Intelligence

Hamed Masoudi

9922102135

Kourosh Sherkat

9912102638

APRIL 21, 2023

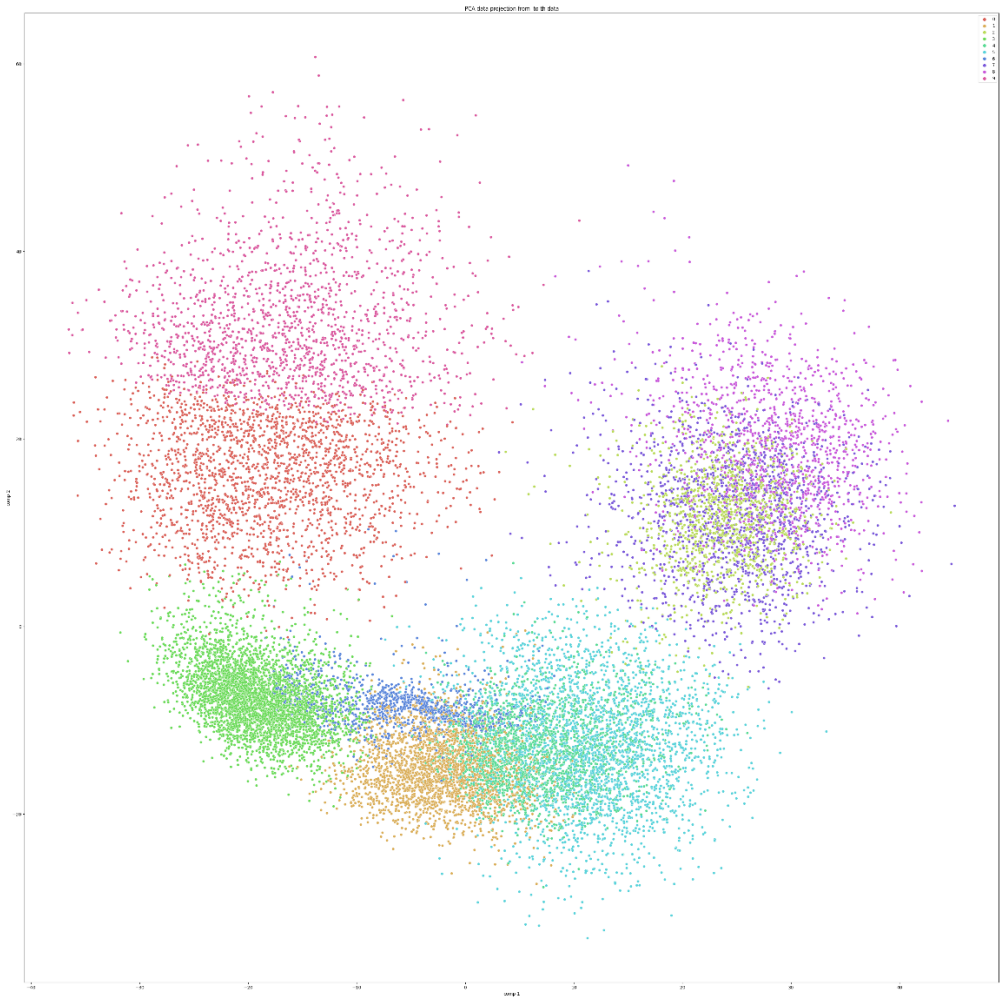
Ferdowsi University of Mashhad
Faculty of Engineering

فاز اول:

در این بخش ما دو الگوریتم Agglomerative با کلاستر ثابت و Kmeans را بررسی کردیم دلیل اینکه Meanshift و Dbscan را در این بررسی وارد نکردیم این بود که برای این دیتا که حجم بسیار زیادی دارد دو الگوریتم Meanshift و Dbscan در سیستم های ما قابل اجرا نبودند.

با استفاده از روش استاندارد سازی، حجم داده ها را افزایش می دهیم تا عمل کلاسترینگ بهتر صورت بگیرد. به این منظور باید هم بر روی داده های ترین و هم داده های تست در بخش لیبل و دامین نیز این عمل را تکرار کنیم تا دقت درستی از مدل دریافت کنیم.

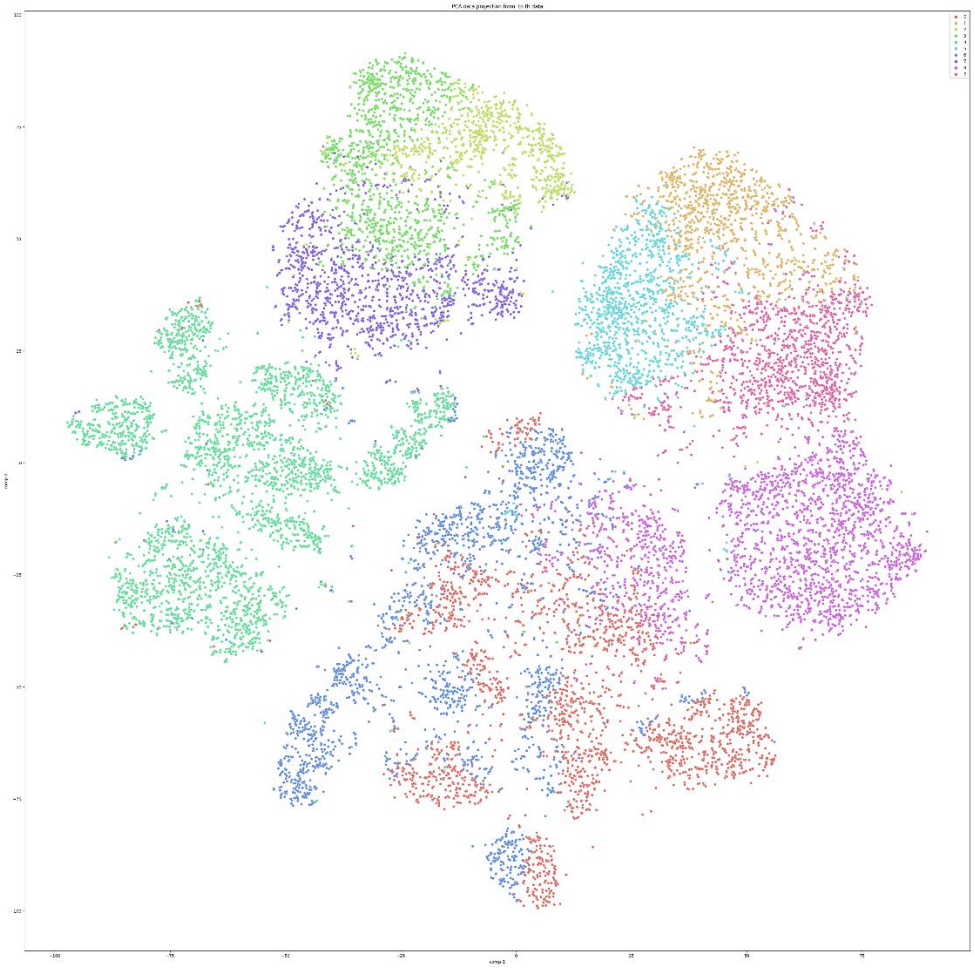
بررسی را با Kmeans با کلاستر ده شروع می کنیم و دو راهکار بصری (T-Sne,PCA) را بررسی می کنیم تا بتوانیم پارامتر تعداد کلاس در Kmeans را به درستی تعیین کنید.



شکل 1. الگوریتم Kmeans با K =10 و استفاده از روش PCA

همانطور که در شکل بالا مشاهده می کنید ما تصمیم گرفتیم که از K = 10 با استفاده کنیم زیرا از لحاظ بصری با PCA بهترین کلاسترینگ را میداد هرچند که شاید K کمتر از لحاظ بصری کلاسترینگ بهتری انجام میداد ولی ما نمی خواستیم درگیر مشکل

Under Clustering بشویم به همین دلیل Over Clustering انجام دادیم تا در مرحله بعدی بتوانیم با استفاده از معیار هایی مانند معیار شباهت تعداد کلاستر ها را تا حد ممکن کاهش بدهیم.

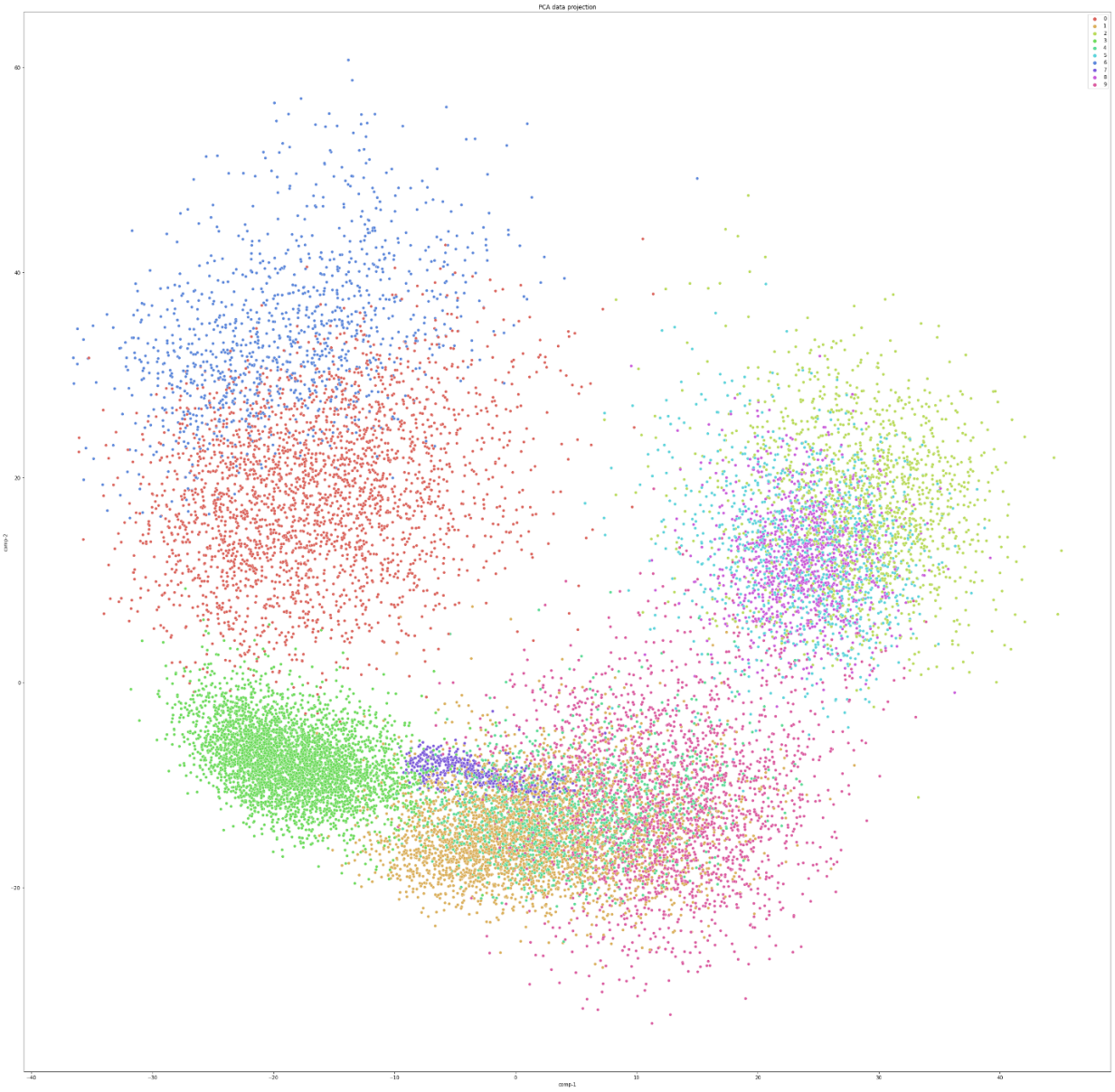


شکل 2. الگوریتم Kmeans با K = 10 و روش بصری سازی Tsne

همانطور که مشخص است K =10 توانستیم نسبتا به درستی عمل کلاسترینگ را انجام بدهیم و در مرحله بعد می توانیم کلاستر هایی که خیلی بهم نزدیک هستند را یک کلاستر در نظر بگیریم تا بتوانیم دقت خودمون را افزایش بدهیم با این ایده مطمئن هستیم که تعداد کلاستر هامون به عدد درستی می رسد اما اگر الان پارامتر K را عدد کمی انتخاب می کردیم ممکن بود یکسری از کلاستر ها را از دست بدیم حتی با توجه به اینکه از لحاظ بصری به نتایج بهتری برسیم.

حال الگوریتم Agglomerative را برای تسک کلاسترینگ بررسی می کنیم و سپس نتایج را با Kmeans بررسی می کنیم و یک الگوریتم را به عنوان الگوریتم مناسب انتخاب می کنیم.

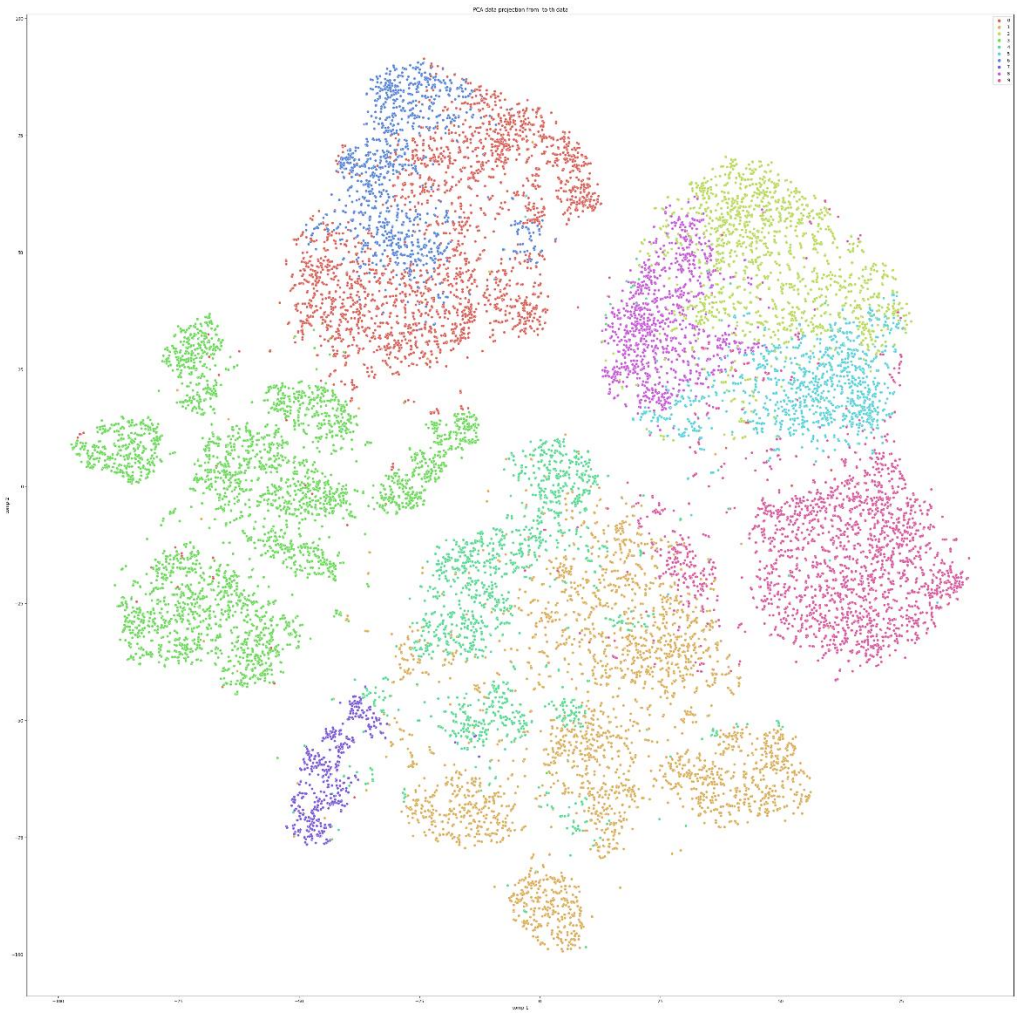
در گام اول ما از الگوریتم Agglomerative با کلاستر ثابت استفاده کردیم سپس از ورژنی استفاده می کنیم که بر اساس distance threshold عمل کلاسترینگ را انجام می دهد.



شکل 3. الگوریتم Agglomerative با K = 10

به همان دلیلی که گفته شد پیش تر تصمیم گرفتیم با Agglomerative هم تعداد کلاستر را بر روی 10 تنظیم کنیم. مشخص است نسبت به الگوریتم Kmeans کیفیت کلاستر ها کمتر است خصوصا در ناحیه پایین تصویر که خطای بسیار زیادی مشاهده می شود از لحاظ بصری هرچند در انتها این بخش ما یکسری متریک را بررسی می کنیم که نیاز به لیبل ندارن و نشان می دهیم که حتی با توجه به آن متریک ها هم Kmeans بهتر عمل کرده است.

حال باید Agglomerative با کلاستر ثابت را با tsne نشان بدهیم.



شکل4 . الگوریتم Agglomerative با کلاستر ثابت و روش بصری سازی Tsne

باز هم مشخص است که الگوریتم Kmeans با تعداد کلاستر 10 بهتر عمل کرده است. حال بر اساس بعضی از متریک ها این ادعا خود را ادامه می دهیم.

ما در این بخش از سه متریک davies_bouldin, silhouette, calinski_harabasz استفاده کردیم نتایج این متریک ها برای Kmeans با K = 10 برابر است با:

Davies Bouldin : 2.50, Silhouette : 0.093, Calinski Harabasz: 2746

مقادیر این سه متریک برای الگوریتم Agglomerative :

Davies Bouldin : 2.7, Silhouette: 0.065, Calinski Harabasz: 2549

همانطور که مشاهده می شود هر سه متریک نشان می دهد که الگوریتم Kmeans برای این مسئله برتری دارد زیرا انتظار داریم که معیار davies اگر مقدار کمتری داشته باشد کیفیت کلاستر بیشتری داشته باشیم که در kmeans کمتر است همچنین دو معیار دیگه هرچه بیشتر باشند نشان می دهند کیفیت کلاستر که بیشتر که هر دو معیار در Kmeans بیشتر هستند.

بخش دوم:

در این قسمت با بررسی متریک‌های استفاده شده در بخش قبل و همچنین با بررسی مدل بصری (Visualization) تصمیم‌گیری برای تعداد کلاستر نهایی را به عمل می آوریم.

ابتدا داده های بخش قبل را مجددا نمایش می دهیم:

Davies Bouldin index	Silhouette Score	Calinski Harabaz Index
2.5072408371094785	0.093733035	2746.6954192327235

جدول 1: جدول اطلاعات معیار های داده شده برای مدل Kmeans با 10 کلاستر

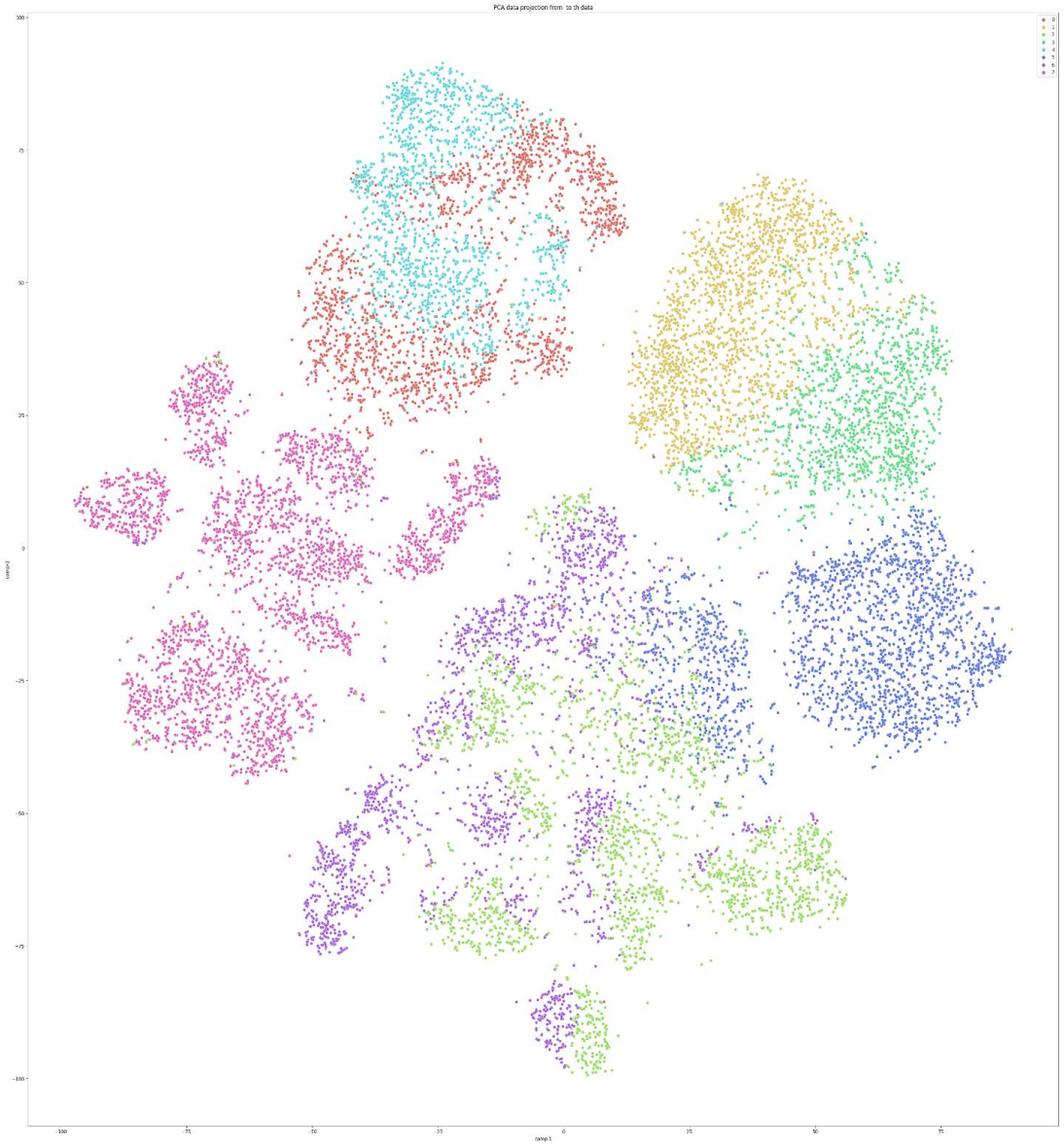
با توجه به توضیحات داده شده در بخش قبل،داده های بدست آمده برای مدل مقادیر خوبی نیست و نیاز به تصحیح مدل می باشد.
برای این مهم ابتدا تصویری از داده های لیبل زده شده توسط الگوریتم کلاسترینگ با تعداد کلاستر 10 را نمایش می دهیم:

شکل 5: تصویر پلات شده با تعداد کلاستر 10

این کلاسترینگ به خوبی داده ها را از هم جدا نکرده و می توان کلاسترها را با یکدیگر ترکیب کنیم تا کلاسترینگ بهتری داشته باشیم.در هر مرحله با ترکیب کلاسترها با هم و بدست آوردن مقادیر متریک های داده شده در جدول 1،کیفیت کلاسترینگ را مشخص می کنیم.برای بهتر شدن کلاسترینگ تعداد کلاسترها را به 8 عدد کاهش می دهیم.نتایج به صورت زیر می شود:

Davies Bouldin index	Silhouette Score	Calinski Harabaz Index
1.906692393049477	0.17058288	4861.048254998224

جدول 6: نتایج بدست آمده برای مدل Kmeans با تعداد کلاستر برابر 8



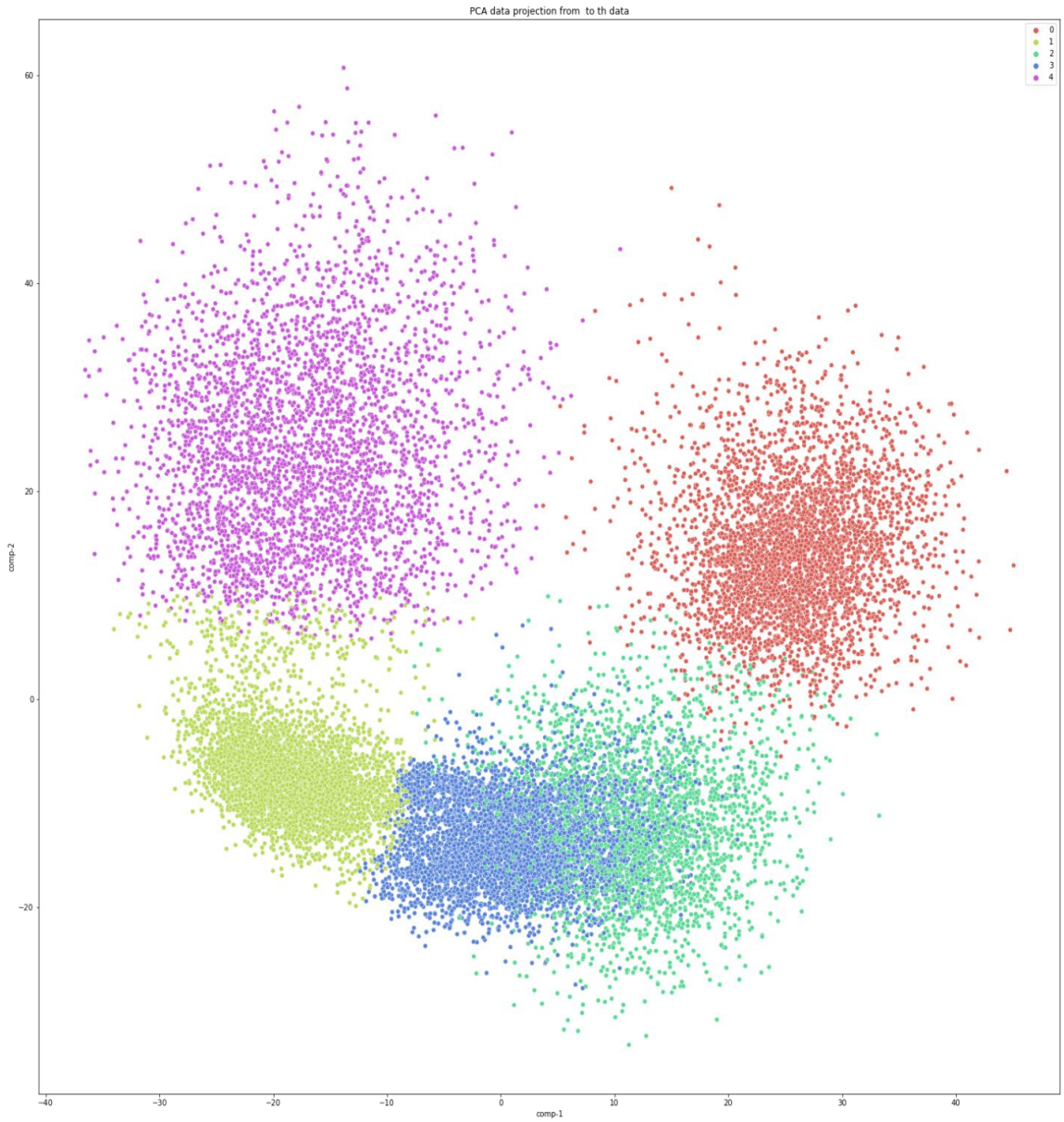
شکل 2: t-SNE برای 8 کلاستر

این نتایج را نیز با ترکیب سه خوشه می توانیم بهبود ببخشیم و در نهایت با K=5 به نتایج بهتری دست بیابیم:

Davies Bouldin index	Silhouette Score	Calinski Harabaz Index
1.8926173501392949	0.17356539	4893.437031576824

جدول 3: جدول نتایج معیارهای ارزیابی برای K=5

و شکل خوشه بندی آن به صورت زیر خواهد شد:



شکل 3: نمایش خوشه بندی با استفاده از متود PCA

در نهایت می توان این نتیجه گیری را به عمل آورد که تعداد دامین های موجود در این دیتاست با تقریب قابل قبولی برابر با 5 می باشد.

بخش سوم:

در این بخش به ارزیابی خوشه بندی انجام شده و همچنین ارزیابی لیبل های هر عکس می پردازیم. برای این کار از تکه کد زیر استفاده می کنیم:

```
from sklearn.metrics import accuracy_score
cluster_labels = []
x_pred = model.predict(st.fit_transform(train_features))
for i in np.unique(x_pred):
   c = (x_pred == i)
   label, count = np.unique(image_labels[c], return_counts=True)
   cluster_labels.append(label[np.argmax(count)])
print(cluster_labels)
test = model.predict(train_features)
x2 = test.copy()
y_pred = np.select([x_pred == i for i in np.unique(x_pred)],cluster_labels,x_pred)
print(f"Accuracy: {accuracy_score(image_labels,y_pred) * 100}%")
```

کد بالا با بررسی مقادیر در هر خوشه،شماره ی متناظر آن در لیبل دامین ها را به خوشه نسبت می دهد و در نهایت با تعویض لیبل خوشه با لیبل منتسب، مقدار دقت مدل را می سنجد.

خروجی کد بالا به صورت زیر خواهد بود:

```
[4, 0, 2, 1, 3]
Accuracy: 89.2%
```

که بیناگر این می باشد که دقت خوشه بندی انجام شده در حدود 89 درصد می باشد.

برای بررسی دقت مدل در پیشبینی لیبل هر تصویر از تکه کد زیر استفاده می کنیم:

```
test = st.fit_transform(test_features)
scores = np.empty([5,])
x_pred = model.predict(test)

for i in range(5):
   scores[i] = accuracy_score(iMage_labels[x_pred == i] , x_pred[x_pred == i])
print(f"Image Label Accuracy: {scores.mean() * 100}%")
```

که خروجی آن نیز به صورت زیر خواهد بود:

```
Image Label Accuracy: 12.094619723060596%
```

این بدین معناست که دقت مدل خوشه بندی آموزش داده شده برای پیشبینی لیبل تصاویر مناسب نمی باشد و نمی توان به از آن به عنوان یک مدل برای پیشبینی لیبل تصویر استفاده نمود.