**Unsupervised Learning on Mice Genetics Dataset**

**Objective**

The objective of this project is to apply unsupervised learning algorithms to the Mice Genetics dataset to predict clusters and compare them with the target labels. The goal is to identify the best clustering model that aligns well with the known categories.

**Description of the Data**

The Mice Genetics dataset consists of 80 columns, with 3 non-numerical columns. The dataset includes various protein expression levels measured in mice, which are used to identify different genetic backgrounds or conditions.

**Data Exploration and Cleaning**

1. **Initial Exploration**:
   * The dataset was initially explored to understand the distribution and types of data.
   * Non-numerical columns were identified and excluded from the analysis.
2. **Handling Missing Values**:
   * All rows with missing values (NA) were dropped to simplify the analysis.
3. **Principal Component Analysis (PCA)**:
   * PCA was performed to reduce the dimensionality of the data to 2 components for easier visualization and clustering.
   * The PCA-transformed data was used for fitting the clustering models.

**Summary of Different Models Trained**

Three unsupervised learning algorithms were applied to the PCA-transformed data:

1. **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**:
   * **Parameter Search**: A for loop was used to search for the best parameters (eps and min\_samples).
   * **Performance**: DBSCAN performed the best among the three models, effectively identifying clusters and handling noise points.
2. **Agglomerative Clustering**:
   * **Parameter Search**: Different linkage methods and the number of clusters were tested.
   * **Performance**: Agglomerative clustering provided reasonable results but was less effective than DBSCAN.
3. **K-Means Clustering**:
   * **Parameter Search**: The number of clusters (K) was plotted using the elbow method to find the optimal value which was 4.
   * **Performance**: K-Means clustering was straightforward but did not perform as well as DBSCAN in capturing the underlying structure of the data.

**Best Model: DBSCAN**

* **Best Parameters**: The optimal parameters for DBSCAN were found to be eps=0.7 and min\_samples=2.
* **Evaluation Metrics**:
  + **Silhouette Score**: 0.60

**Final Results**

* **Clusters Identified**: DBSCAN successfully identified meaningful clusters in the PCA-transformed data.
* **Comparison with Target Labels**: The clusters were compared with the target labels using Silhouette Score, showing a strong alignment.
* **Visualization**: The clusters were visualized using scatter plots, highlighting the separation and structure of the clusters

**!**pip install ucimlrepo

**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**%matplotlib** inline

**from** ucimlrepo **import** fetch\_ucirepo

*# fetch dataset*

mice\_protein\_expression **=** fetch\_ucirepo(id**=**342)

*# data (as pandas dataframes)*

X **=** mice\_protein\_expression**.**data**.**features

y **=** mice\_protein\_expression**.**data**.**targets

*# metadata*

print(mice\_protein\_expression**.**metadata)

*# variable information*

print(mice\_protein\_expression**.**variables)

{'uci\_id': 342, 'name': 'Mice Protein Expression', 'repository\_url': 'https://archive.ics.uci.edu/dataset/342/mice+protein+expression', 'data\_url': 'https://archive.ics.uci.edu/static/public/342/data.csv', 'abstract': 'Expression levels of 77 proteins measured in the cerebral cortex of 8 classes of control and Down syndrome mice exposed to context fear conditioning, a task used to assess associative learning.', 'area': 'Biology', 'tasks': ['Classification', 'Clustering'], 'characteristics': ['Multivariate'], 'num\_instances': 1080, 'num\_features': 80, 'feature\_types': ['Real'], 'demographics': [], 'target\_col': ['class'], 'index\_col': ['MouseID'], 'has\_missing\_values': 'yes', 'missing\_values\_symbol': 'NaN', 'year\_of\_dataset\_creation': 2015, 'last\_updated': 'Tue Apr 16 2024', 'dataset\_doi': '10.24432/C50S3Z', 'creators': ['Clara Higuera', 'Katheleen Gardiner', 'Krzysztof Cios'], 'intro\_paper': {'title': 'Self-Organizing Feature Maps Identify Proteins Critical to Learning in a Mouse Model of Down Syndrome', 'authors': 'C. Higuera, K. Gardiner, K. Cios', 'published\_in': 'PLoS ONE', 'year': 2015, 'url': 'https://www.semanticscholar.org/paper/5c5754b02a4f2f36ccf8cdda78059cdb19860532', 'doi': '10.1371/journal.pone.0129126'}, 'additional\_info': {'summary': 'The data set consists of the expression levels of 77 proteins/protein modifications that produced detectable signals in the nuclear fraction of cortex. There are 38 control mice and 34 trisomic mice (Down syndrome), for a total of 72 mice. In the experiments, 15 measurements were registered of each protein per sample/mouse. Therefore, for control mice, there are 38x15, or 570 measurements, and for trisomic mice, there are 34x15, or 510 measurements. The dataset contains a total of 1080 measurements per protein. Each measurement can be considered as an independent sample/mouse.\r\n\r\nThe eight classes of mice are described based on features such as genotype, behavior and treatment. According to genotype, mice can be control or trisomic. According to behavior, some mice have been stimulated to learn (context-shock) and others have not (shock-context) and in order to assess the effect of the drug memantine in recovering the ability to learn in trisomic mice, some mice have been injected with the drug and others have not.\r\n\r\nClasses:\r\nc-CS-s: control mice, stimulated to learn, injected with saline (9 mice)\r\nc-CS-m: control mice, stimulated to learn, injected with memantine (10 mice)\r\nc-SC-s: control mice, not stimulated to learn, injected with saline (9 mice)\r\nc-SC-m: control mice, not stimulated to learn, injected with memantine (10 mice)\r\n\r\nt-CS-s: trisomy mice, stimulated to learn, injected with saline (7 mice)\r\nt-CS-m: trisomy mice, stimulated to learn, injected with memantine (9 mice)\r\nt-SC-s: trisomy mice, not stimulated to learn, injected with saline (9 mice)\r\nt-SC-m: trisomy mice, not stimulated to learn, injected with memantine (9 mice)\r\n\r\nThe aim is to identify subsets of proteins that are discriminant between the classes.\r\n', 'purpose': None, 'funded\_by': None, 'instances\_represent': None, 'recommended\_data\_splits': None, 'sensitive\_data': None, 'preprocessing\_description': None, 'variable\_info': '1 Mouse ID \r\n2..78 Values of expression levels of 77 proteins; the names of proteins are followed by â€œ\_nâ€\x9d indicating that they were measured in the nuclear fraction. For example: DYRK1A\_n\r\n79 Genotype: control (c) or trisomy (t)\r\n80 Treatment type: memantine (m) or saline (s)\r\n81 Behavior: context-shock (CS) or shock-context (SC)\r\n82 Class: c-CS-s, c-CS-m, c-SC-s, c-SC-m, t-CS-s, t-CS-m, t-SC-s, t-SC-m \r\n', 'citation': None}}

name role type ... description units missing\_values

0 MouseID ID Categorical ... None None no

1 DYRK1A\_N Feature Continuous ... None None yes

2 ITSN1\_N Feature Continuous ... None None no

3 BDNF\_N Feature Continuous ... None None yes

4 NR1\_N Feature Continuous ... None None no

.. ... ... ... ... ... ... ...

77 CaNA\_N Feature Continuous ... None None no

78 Genotype Feature Categorical ... None None no

79 Treatment Feature Categorical ... None None no

80 Behavior Feature Categorical ... None None no

81 class Target Categorical ... None None no

[82 rows x 7 columns]

X**.**head()

y

|  | **class** |
| --- | --- |
| **0** | c-CS-m |
| **1** | c-CS-m |
| **2** | c-CS-m |
| **3** | c-CS-m |
| **4** | c-CS-m |
| **...** | ... |
| **1075** | t-SC-s |
| **1076** | t-SC-s |
| **1077** | t-SC-s |
| **1078** | t-SC-s |
| **1079** | t-SC-s |

1080 rows × 1 columns

X**.**describe()

X**.**info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1080 entries, 0 to 1079

Data columns (total 80 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 DYRK1A\_N 1077 non-null float64

1 ITSN1\_N 1077 non-null float64

2 BDNF\_N 1077 non-null float64

3 NR1\_N 1077 non-null float64

4 NR2A\_N 1077 non-null float64

5 pAKT\_N 1077 non-null float64

6 pBRAF\_N 1077 non-null float64

7 pCAMKII\_N 1077 non-null float64

8 pCREB\_N 1077 non-null float64

9 pELK\_N 1077 non-null float64

10 pERK\_N 1077 non-null float64

11 pJNK\_N 1077 non-null float64

12 PKCA\_N 1077 non-null float64

13 pMEK\_N 1077 non-null float64

14 pNR1\_N 1077 non-null float64

15 pNR2A\_N 1077 non-null float64

16 pNR2B\_N 1077 non-null float64

17 pPKCAB\_N 1077 non-null float64

18 pRSK\_N 1077 non-null float64

19 AKT\_N 1077 non-null float64

20 BRAF\_N 1077 non-null float64

21 CAMKII\_N 1077 non-null float64

22 CREB\_N 1077 non-null float64

23 ELK\_N 1062 non-null float64

24 ERK\_N 1077 non-null float64

25 GSK3B\_N 1077 non-null float64

26 JNK\_N 1077 non-null float64

27 MEK\_N 1073 non-null float64

28 TRKA\_N 1077 non-null float64

29 RSK\_N 1077 non-null float64

30 APP\_N 1077 non-null float64

31 Bcatenin\_N 1062 non-null float64

32 SOD1\_N 1077 non-null float64

33 MTOR\_N 1077 non-null float64

34 P38\_N 1077 non-null float64

35 pMTOR\_N 1077 non-null float64

36 DSCR1\_N 1077 non-null float64

37 AMPKA\_N 1077 non-null float64

38 NR2B\_N 1077 non-null float64

39 pNUMB\_N 1077 non-null float64

40 RAPTOR\_N 1077 non-null float64

41 TIAM1\_N 1077 non-null float64

42 pP70S6\_N 1077 non-null float64

43 NUMB\_N 1080 non-null float64

44 P70S6\_N 1080 non-null float64

45 pGSK3B\_N 1080 non-null float64

46 pPKCG\_N 1080 non-null float64

47 CDK5\_N 1080 non-null float64

48 S6\_N 1080 non-null float64

49 ADARB1\_N 1080 non-null float64

50 AcetylH3K9\_N 1080 non-null float64

51 RRP1\_N 1080 non-null float64

52 BAX\_N 1080 non-null float64

53 ARC\_N 1080 non-null float64

54 ERBB4\_N 1080 non-null float64

55 nNOS\_N 1080 non-null float64

56 Tau\_N 1080 non-null float64

57 GFAP\_N 1080 non-null float64

58 GluR3\_N 1080 non-null float64

59 GluR4\_N 1080 non-null float64

60 IL1B\_N 1080 non-null float64

61 P3525\_N 1080 non-null float64

62 pCASP9\_N 1080 non-null float64

63 PSD95\_N 1080 non-null float64

64 SNCA\_N 1080 non-null float64

65 Ubiquitin\_N 1080 non-null float64

66 pGSK3B\_Tyr216\_N 1080 non-null float64

67 SHH\_N 1080 non-null float64

68 BAD\_N 867 non-null float64

69 BCL2\_N 795 non-null float64

70 pS6\_N 1080 non-null float64

71 pCFOS\_N 1005 non-null float64

72 SYP\_N 1080 non-null float64

73 H3AcK18\_N 900 non-null float64

74 EGR1\_N 870 non-null float64

75 H3MeK4\_N 810 non-null float64

76 CaNA\_N 1080 non-null float64

77 Genotype 1080 non-null object

78 Treatment 1080 non-null object

79 Behavior 1080 non-null object

dtypes: float64(77), object(3)

memory usage: 675.1+ KB

X **=** X**.**drop(['Genotype','Treatment','Behavior'],axis**=**1)

X**.**head()

X **=** X**.**dropna()

X**.**isna()**.**sum()

Out[12]:

DYRK1A\_N 0

ITSN1\_N 0

BDNF\_N 0

NR1\_N 0

NR2A\_N 0

..

SYP\_N 0

H3AcK18\_N 0

EGR1\_N 0

H3MeK4\_N 0

CaNA\_N 0

Length: 77, dtype: int64

**from** sklearn.decomposition **import** PCA

**from** sklearn.preprocessing **import** OneHotEncoder

pca **=** PCA(n\_components**=**2)

pca\_result **=** pca**.**fit\_transform(X)

plt**.**figure(figsize**=**(8, 6))

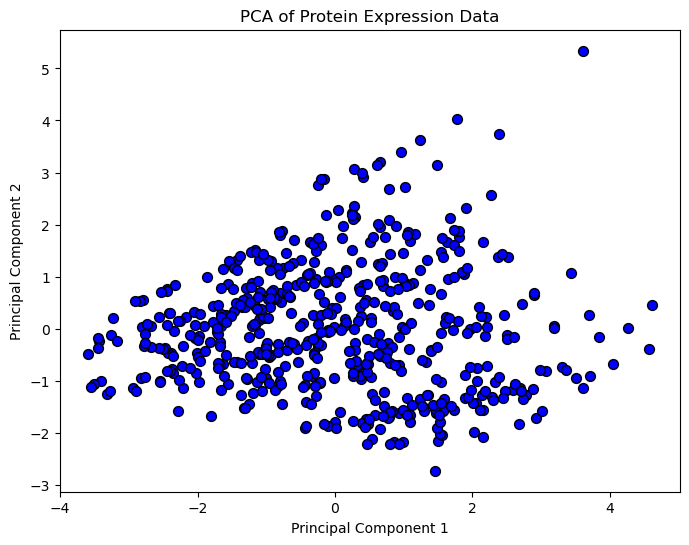
plt**.**scatter(pca\_result[:, 0], pca\_result[:, 1], c**=**'blue', edgecolor**=**'k', s**=**50)

plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

plt**.**title('PCA of Protein Expression Data')

plt**.**show()



**from** sklearn.cluster **import** KMeans

kmeans **=** KMeans(n\_clusters**=**4)

kmeans**.**fit(pca\_result)

cluster\_labels **=** kmeans**.**labels\_

plt**.**figure(figsize**=**(8, 6))

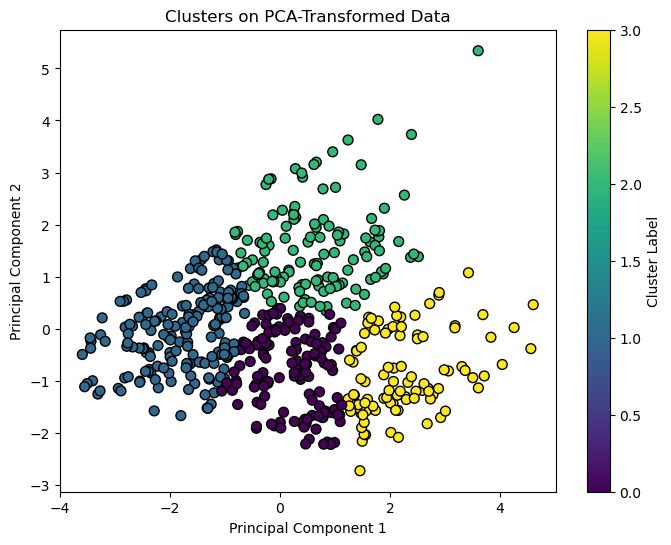
plt**.**scatter(pca\_result[:, 0], pca\_result[:, 1], c**=**cluster\_labels, cmap**=**'viridis', edgecolor**=**'k', s**=**50)

plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

plt**.**title('Clusters on PCA-Transformed Data')

plt**.**colorbar(label**=**'Cluster Label')

plt**.**show()

sse **=** []

**for** k **in** range(1, 11):

kmeans **=** KMeans(n\_clusters**=**k)

kmeans**.**fit(pca\_result)

sse**.**append(kmeans**.**inertia\_)

plt**.**figure(figsize**=**(8, 6))

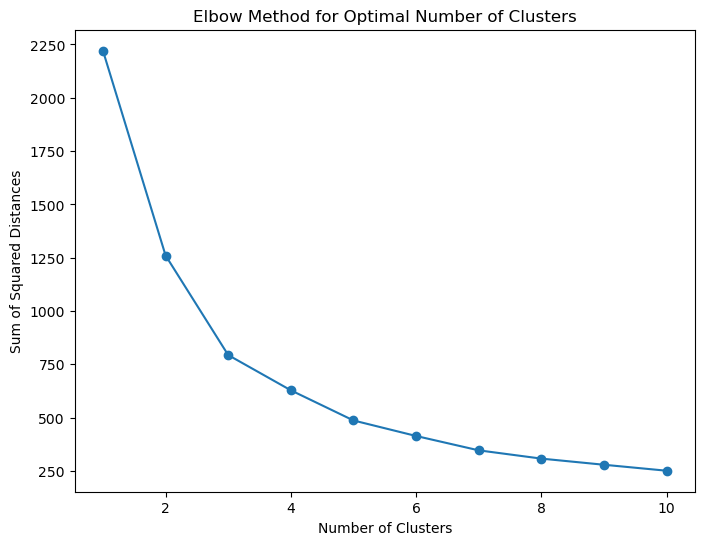
plt**.**plot(range(1, 11), sse, marker**=**'o')

plt**.**xlabel('Number of Clusters')

plt**.**ylabel('Sum of Squared Distances')

plt**.**title('Elbow Method for Optimal Number of Clusters')

plt**.**show()



**from** sklearn.cluster **import** DBSCAN

**from** sklearn.model\_selection **import** GridSearchCV

**from** sklearn.metrics **import** silhouette\_score

**from** sklearn.model\_selection **import** KFold,ParameterGrid

**from** sklearn.metrics **import** adjusted\_rand\_score, normalized\_mutual\_info\_score

param\_grid **=** {

'eps': np**.**arange(0.1, 1.0, 0.1),

'min\_samples': range(2, 10)

}

best\_params **=** **None**

best\_score **=** **-**1

**for** params **in** ParameterGrid(param\_grid):

dbscan **=** DBSCAN(eps**=**params['eps'], min\_samples**=**params['min\_samples'])

labels **=** dbscan**.**fit\_predict(pca\_result)

**if** len(set(labels)) **>** 1:

score **=** silhouette\_score(pca\_result, labels)

**if** score **>** best\_score:

best\_score **=** score

best\_params **=** params

print(f'Best Parameters: {best\_params}')

print(f'Best Silhouette Score: {best\_score}')

dbscan **=** DBSCAN(eps**=**best\_params['eps'], min\_samples**=**best\_params['min\_samples'])

labels **=** dbscan**.**fit\_predict(pca\_result)

Best Parameters: {'eps': 0.7000000000000001, 'min\_samples': 2}

Best Silhouette Score: 0.6000363590678445

plt**.**figure(figsize**=**(8, 6))

plt**.**scatter(pca\_result[:, 0], pca\_result[:, 1], c**=**labels, cmap**=**'viridis', edgecolor**=**'k', s**=**50)

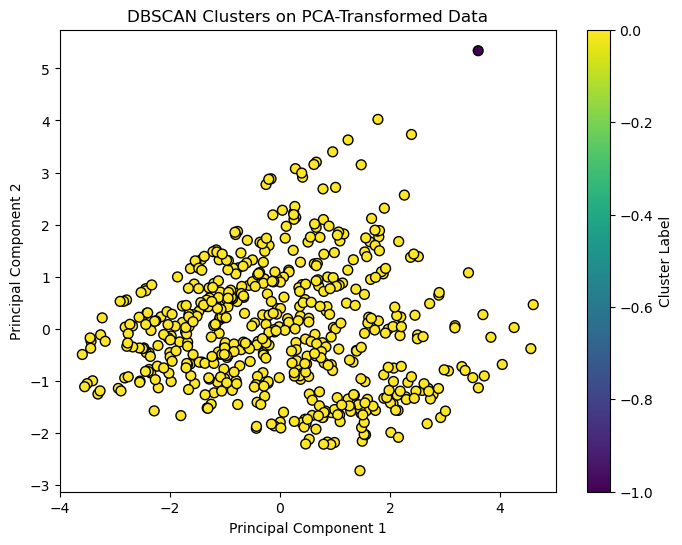
plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

plt**.**title('DBSCAN Clusters on PCA-Transformed Data')

plt**.**colorbar(label**=**'Cluster Label')

plt**.**show()



**from** sklearn.cluster **import** AgglomerativeClustering

**from** scipy.cluster.hierarchy **import** dendrogram, linkage

hierarchical **=** AgglomerativeClustering(n\_clusters**=**3,linkage**=**'ward')

hierarchical**.**fit(pca\_result)

hierarchical\_labels **=** hierarchical**.**labels\_

**from** sklearn.metrics **import** silhouette\_score, davies\_bouldin\_score

silhouette\_avg **=** silhouette\_score(pca\_result, hierarchical\_labels)

db\_index **=** davies\_bouldin\_score(pca\_result, hierarchical\_labels)

print(f'Silhouette Score: {silhouette\_avg}')

print(f'Davies-Bouldin Index: {db\_index}')

Silhouette Score: 0.3728050389923807

Davies-Bouldin Index: 0.8671802038921039

param\_grid **=** {

'n\_clusters': [2, 3, 4, 5],

'linkage': ['ward', 'complete', 'average', 'single']

}

best\_params **=** **None**

best\_score **=** **-**1

kf **=** KFold(n\_splits**=**5, shuffle**=True**, random\_state**=**42)

**for** params **in** ParameterGrid(param\_grid):

scores **=** []

**for** train\_index, val\_index **in** kf**.**split(pca\_result):

train\_data, val\_data **=** pca\_result[train\_index], pca\_result[val\_index]

hierarchical **=** AgglomerativeClustering(n\_clusters**=**params['n\_clusters'], linkage**=**params['linkage'])

hierarchical**.**fit(train\_data)

labels **=** hierarchical**.**fit\_predict(val\_data)

**if** len(set(labels)) **>** 1:

score **=** silhouette\_score(val\_data, labels)

scores**.**append(score)

avg\_score **=** np**.**mean(scores)

**if** avg\_score **>** best\_score:

best\_score **=** avg\_score

best\_params **=** params

print(f'Best Parameters: {best\_params}')

print(f'Best Silhouette Score: {best\_score}')

Best Parameters: {'linkage': 'single', 'n\_clusters': 2}

Best Silhouette Score: 0.4155256983433387

Z **=** linkage(pca\_result, method**=**'ward')

plt**.**figure(figsize**=**(10, 7))

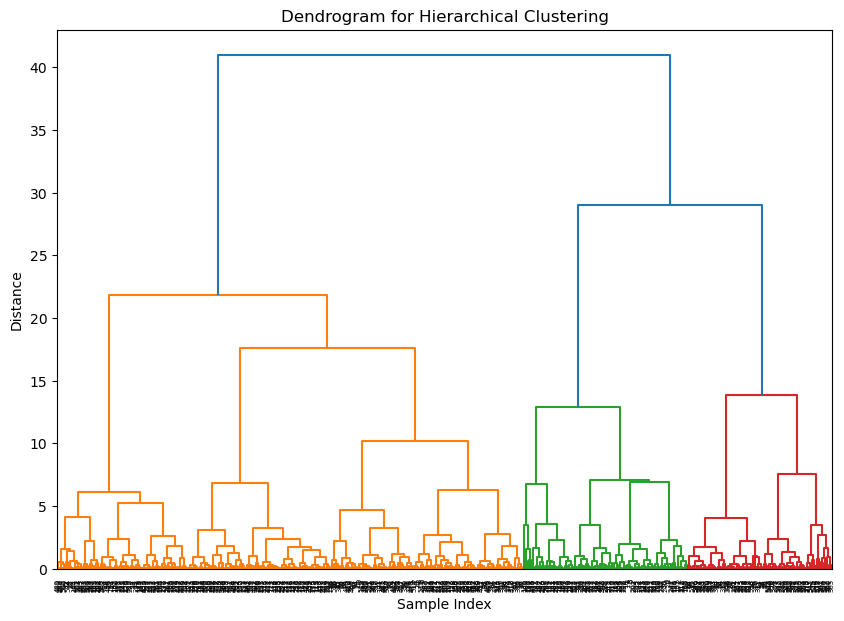
dendrogram(Z)

plt**.**title('Dendrogram for Hierarchical Clustering')

plt**.**xlabel('Sample Index')

plt**.**ylabel('Distance')

plt**.**show()



plt**.**figure(figsize**=**(8, 6))

plt**.**scatter(pca\_result[:, 0], pca\_result[:, 1], c**=**hierarchical\_labels, cmap**=**'viridis', edgecolor**=**'k', s**=**50)

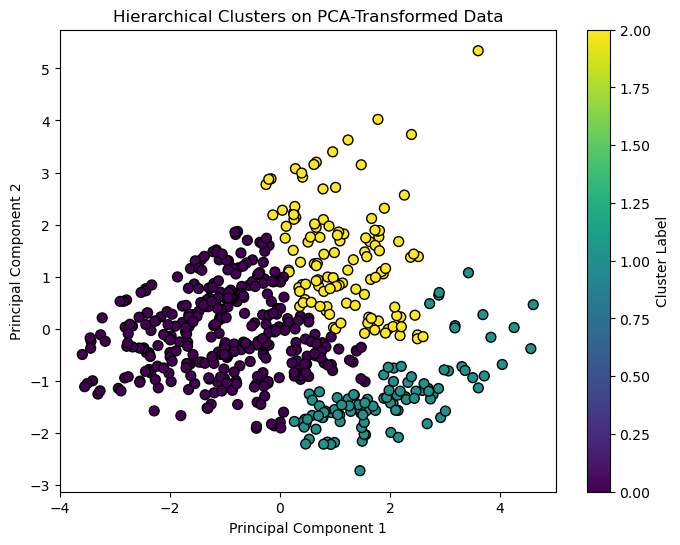
plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

plt**.**title('Hierarchical Clusters on PCA-Transformed Data')

plt**.**colorbar(label**=**'Cluster Label')

plt**.**show()



**for** cluster **in** range(3):

cluster\_data **=** X[hierarchical\_labels **==** cluster]

print(f'Cluster {cluster} Summary:')

print(cluster\_data**.**describe())

Cluster 0 Summary:

DYRK1A\_N ITSN1\_N BDNF\_N ... EGR1\_N H3MeK4\_N CaNA\_N

count 332.000000 332.000000 332.000000 ... 332.000000 332.000000 332.000000

mean 0.391108 0.581345 0.287977 ... 0.180608 0.197657 1.367432

std 0.158436 0.165190 0.038011 ... 0.034452 0.044843 0.324345

min 0.145327 0.245359 0.115181 ... 0.121468 0.101787 0.788462

25% 0.273292 0.456648 0.263937 ... 0.156212 0.164728 1.108069

50% 0.356666 0.558752 0.290665 ... 0.173757 0.191075 1.357109

75% 0.452284 0.677435 0.311170 ... 0.200960 0.219010 1.612432

max 0.992220 1.151506 0.417221 ... 0.326143 0.333271 2.129791

[8 rows x 77 columns]

Cluster 1 Summary:

DYRK1A\_N ITSN1\_N BDNF\_N ... EGR1\_N H3MeK4\_N CaNA\_N

count 103.000000 103.000000 103.000000 ... 103.000000 103.000000 103.000000

mean 0.421010 0.627181 0.334053 ... 0.196199 0.217768 1.232963

std 0.164599 0.180628 0.045435 ... 0.041969 0.044110 0.238048

min 0.194417 0.348331 0.232139 ... 0.142407 0.146804 0.872258

25% 0.290289 0.502910 0.301343 ... 0.168336 0.186181 1.035062

50% 0.365907 0.577851 0.324419 ... 0.186439 0.210056 1.171313

75% 0.539343 0.715363 0.361587 ... 0.214329 0.238219 1.455418

max 0.940956 1.085552 0.443358 ... 0.360692 0.372005 1.696719

[8 rows x 77 columns]

Cluster 2 Summary:

DYRK1A\_N ITSN1\_N BDNF\_N ... EGR1\_N H3MeK4\_N CaNA\_N

count 117.000000 117.000000 117.000000 ... 117.000000 117.000000 117.000000

mean 0.478623 0.738057 0.374891 ... 0.156796 0.162574 1.537491

std 0.160074 0.183585 0.039257 ... 0.019589 0.029317 0.330924

min 0.256346 0.477769 0.303897 ... 0.120911 0.115270 0.795637

25% 0.334410 0.592269 0.344816 ... 0.142573 0.141726 1.294631

50% 0.456786 0.719069 0.373247 ... 0.155666 0.159211 1.562096

75% 0.573992 0.857961 0.397923 ... 0.167615 0.179059 1.806884

max 0.945885 1.336398 0.497160 ... 0.226396 0.284145 2.115555

[8 rows x 77 columns]

eps\_values **=** [0.3, 0.5, 0.7]

min\_samples\_values **=** [3, 5, 7]

best\_ari **=** **-**1

best\_params **=** **None**

**for** eps **in** eps\_values:

**for** min\_samples **in** min\_samples\_values:

*# Perform DBSCAN clustering*

dbscan **=** DBSCAN(eps**=**eps, min\_samples**=**min\_samples)

dbscan\_labels **=** dbscan**.**fit\_predict(pca\_result)

*# Convert labels to a DataFrame*

labels\_df **=** pd**.**DataFrame({'Cluster': dbscan\_labels})

*# Merge clustering results with target labels*

results **=** pd**.**concat([labels\_df, y**.**reset\_index(drop**=True**)], axis**=**1)

*# Handle noise points (label -1)*

results['Cluster'] **=** results['Cluster']**.**astype(str)

*# Evaluate clustering performance*

ari **=** adjusted\_rand\_score(y['class'], results['Cluster'])

nmi **=** normalized\_mutual\_info\_score(y['class'], results['Cluster'])

silhouette\_avg **=** silhouette\_score(pca\_result, dbscan\_labels)

print(f'eps: {eps}, min\_samples: {min\_samples}, ARI: {ari}, NMI: {nmi}, Silhouette Score: {silhouette\_avg}')

**if** ari **>** best\_ari:

best\_ari **=** ari

best\_params **=** (eps, min\_samples)

print(f'Best Parameters: eps={best\_params[0]}, min\_samples={best\_params[1]}')

print(f'Best Adjusted Rand Index: {best\_ari}')

eps: 0.3, min\_samples: 3, ARI: 0.22479396609453056, NMI: 0.4371835273504064, Silhouette Score: -0.08980357358168613

eps: 0.3, min\_samples: 5, ARI: 0.2190201810957853, NMI: 0.4207028158768853, Silhouette Score: -0.027080951387885178

eps: 0.3, min\_samples: 7, ARI: 0.21832257727878246, NMI: 0.4138516040332791, Silhouette Score: 0.001829309894662775

eps: 0.5, min\_samples: 3, ARI: 0.23384951320847377, NMI: 0.45367233055238393, Silhouette Score: 0.17924611069784274

eps: 0.5, min\_samples: 5, ARI: 0.23336832624574125, NMI: 0.45027120460913, Silhouette Score: 0.23457737596562436

eps: 0.5, min\_samples: 7, ARI: 0.23306053996224788, NMI: 0.45001363774247477, Silhouette Score: 0.23647298710010312

eps: 0.7, min\_samples: 3, ARI: 0.23620823173264943, NMI: 0.4650033580452026, Silhouette Score: 0.6000363590678445

eps: 0.7, min\_samples: 5, ARI: 0.235810802852644, NMI: 0.4649984220868739, Silhouette Score: 0.5027773495899465

eps: 0.7, min\_samples: 7, ARI: 0.23582809114982572, NMI: 0.46330664825986206, Silhouette Score: 0.47893138355358683

Best Parameters: eps=0.7, min\_samples=3

Best Adjusted Rand Index: 0.23620823173264943

dbscan **=** DBSCAN(eps**=**0.7, min\_samples**=**2)

dbscan\_labels **=** dbscan**.**fit\_predict(pca\_result)

In [36]:

plt**.**figure(figsize**=**(8, 6))

plt**.**scatter(pca\_result[:, 0], pca\_result[:, 1], c**=**dbscan\_labels, cmap**=**'viridis', edgecolor**=**'k', s**=**50)

plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

plt**.**title('DBSCAN Clusters on PCA-Transformed Data')

plt**.**colorbar(label**=**'Cluster Label')

plt**.**show()

