1 Import Libraries

In [1]:

```
import numpy as np
import pandas as pd
import pickle
from sklearn.preprocessing import StandardScaler
np.random.seed(22) # for consistency
```

2 Principal Component Analysis

2.1 Objective

Digunakan untuk mengidentifikasi pola pada data dan menemukan pola untuk mereduksi dimensi pada dataset dengan kehilangan informasi yang seminimal mungkin

2.2 Algorithm for dimensionality reduction

1. Load atau ambil keseluruhan dataset yang terdiri atas *d*-dimensional atribut dengan membuang atribut label kelas
2. Hitung covariance matrix dari dataset tersebut
3. Hitung eigen vektor dengan masing-masing eigen value yang bersesuaian
4. Urutkan eigen vector dengan mengurutkan nilai eigen value secara decreasing
5. Pilih k eigen vector dengan beberapa eigen value terbesar pertama untuk membentuk *dxk*-dimensional matrix **W** (dimana setiap kolom *k* yang terbentuk merepresentasikan eigen vector)
6. Gunakan matrix **W** untuk mentransformasi sampel dataset ke dalam ranah PCA yang baru dengan persamaan

y = W.T x D

- W : matriks W
- T : transpose matrix
- D : satu sampel data (*d x 1-dimensional*)
- y : sampel D hasil transformasi PCA

2.3 PCA Example Calculation for Dummy Data

2.3.1 Generating 3 dimensional dummy data

20 sampel data berlabel 0
20 sampel data berlabel 1
Representasi Data Dummy

$$\boldsymbol{X} = \begin{pmatrix} x_{1_1} & x_{1_2} & \dots & x_{1_{20}} \\ x_{2_1} & x_{2_2} & \dots & x_{2_{20}} \\ x_{3_1} & x_{3_2} & \dots & x_{3_{20}} \end{pmatrix}$$

In [2]:

```
mu_vec1 = np.array([0,0,0])
cov_mat1 = np.array([[1,0,0],[0,1,0],[0,0,1]])
class1_sample = np.random.multivariate_normal(mu_vec1, cov_mat1, 20).T
```

In [3]:

```
mu_vec2 = np.array([1,1,1])
cov_mat2 = np.array([[1,0,0],[0,1,0],[0,0,1]])
class2_sample = np.random.multivariate_normal(mu_vec2, cov_mat2, 20).T
```

In [4]:

```
class1_sample.shape
```

Out[4]:

```
(3, 20)
```

In [41]:

```
pd.DataFrame(class1_sample)
```

Out[41]:

	0	1	2	3	4	5	6	7	
0	-0.091950	-0.239325	0.918822	-0.561514	0.587752	1.055972	1.520130	-1.598613	1.04672
1	-1.463351	-0.491129	-1.103632	0.028855	0.752318	0.747750	-1.488603	-0.646074	0.62914
2	1.081792	-1.002272	0.626493	-0.230767	-1.058503	1.064677	1.859990	0.337325	0.36305

In [5]:

```
class2_sample.shape
```

Out[5]:

```
(3, 20)
```

In [42]:

```
pd.DataFrame(class2_sample)
```

Out[42]:

	0	1	2	3	4	5	6	7	8
0	1.012826	3.516222	0.685177	1.950711	-0.376057	1.535522	-0.179403	1.980238	-0.248592
1	-0.231716	-1.039029	1.490315	1.760372	0.728445	2.159998	0.454701	2.006888	0.576167
2	-0.053085	1.094487	1.354976	1.011504	1.542252	0.834995	1.270503	1.782156	1.552608

Dummy Data Description

Baris menyatakan dimensi dan kolom menyatakan jumlah data

2.3.2 Load atau ambil keseluruhan dataset yang terdiri atas *d*-dimensional atribut dengan membuang atribut label kelas

In [7]:

```
all_samples = np.concatenate((class1_sample,class2_sample),axis=1)
```

In [8]:

```
pd.DataFrame(all_samples.T)
```

Out[8]:

	0	1	2
0	-0.091950	-1.463351	1.081792
1	-0.239325	-0.491129	-1.002272
2	0.918822	-1.103632	0.626493
3	-0.561514	0.028855	-0.230767
4	0.587752	0.752318	-1.058503
5	1.055972	0.747750	1.064677
6	1.520130	-1.488603	1.859990
7	-1.598613	-0.646074	0.337325
8	1.046729	0.629143	0.363059
9	0.555750	-1.088550	0.023695
10	2.499177	-2.490030	-0.234862
11	-0.097563	-0.886529	-0.136713
12	0.101979	-0.250930	-0.078812
13	-1.085164	0.594928	-0.638908
14	-1.107837	2.106146	-0.567388
15	-0.479950	-1.923230	0.399589
16	-1.048184	-0.693879	0.745339
17	0.536910	-0.732722	0.555716
18	0.432289	-0.135804	-0.941106
19	0.484767	-1.532821	0.404979
20	1.012826	-0.231716	-0.053085
21	3.516222	-1.039029	1.094487
22	0.685177	1.490315	1.354976
23	1.950711	1.760372	1.011504
24	-0.376057	0.728445	1.542252
25	1.535522	2.159998	0.834995
26	-0.179403	0.454701	1.270503
27	1.980238	2.006888	1.782156
28	-0.248592	0.576167	1.552608
29	1.329603	1.863118	-0.227378
30	1.624874	-1.769533	0.507822
31	1.069148	0.649997	1.869345
32	0.783236	1.017118	1.689713
33	0.116857	2.497248	0.353455

	0	1	2
34	1.604491	1.214950	0.576399
35	1.104419	1.311572	0.532937
36	3.200549	-0.013738	1.001183
37	2.128293	1.507971	2.079541
38	-0.532475	0.765870	1.038859
39	2.258423	0.178661	0.944531

2.3.3 Hitung covariance matrix

In [9]:

```
cov_mat = np.cov(all_samples)
```

In [10]:

```
cov_mat
```

Out[10]:

```
array([[1.39443758, 0.02443485, 0.31999599],
       [0.02443485, 1.62099202, 0.17702731],
       [0.31999599, 0.17702731, 0.68953337]])
```

2.3.4 Hitung eigen vector dan eigen value dari dekomposisi covariance matrix

In [11]:

```
eigen_vals, eigen_vecs = np.linalg.eig(cov_mat)
```

In [12]:

```
eigen_vals
```

Out[12]:

```
array([0.54310222, 1.47474151, 1.68711925])
```

In [13]:

```
eigen_vecs
```

Out[13]:

```
array([[ 0.34450986, -0.85952652, -0.37752765],
       [ 0.14453306,  0.44591347, -0.88332971],
       [-0.92758997, -0.24975057, -0.27785158]])
```

2.3.5 Urutkan eigen vector dengan mengurutkan nilai eigen value secara decreasing

In [14]:

```
# buat list yang isinya pasangan eigen value dengan masing-masing eigen vector
eigen_pairs = [
    (np.abs(eigen_vals[i]), eigen_vecs[:,i]) for i in range(len(eigen_vals))
]
```

In [15]:

```
# pasangan eigen value dengan eigen vector sebelum diurutkan
eigen_pairs
```

Out[15]:

```
[(0.5431022182343318, array([ 0.34450986,  0.14453306, -0.92758997])),
 (1.474741505219636, array([-0.85952652,  0.44591347, -0.24975057])),
 (1.6871192541852207, array([-0.37752765, -0.88332971, -0.27785158]))]
```

In [16]:

```
# urutkan tuples eigen_pairs secara descending order berdasarkan eigen value
eigen_pairs.sort(key=lambda x:x[0], reverse=True)
```

In [17]:

```
# pasangan eigen value dengan eigen vector setelah diurutkan
eigen_pairs
```

Out[17]:

```
[(1.6871192541852207, array([-0.37752765, -0.88332971, -0.27785158])),
 (1.474741505219636, array([-0.85952652,  0.44591347, -0.24975057])),
 (0.5431022182343318, array([ 0.34450986,  0.14453306, -0.92758997]))]
```

2.3.6 Pilih k eigen vector dengan beberapa eigen value terbesar pertama untuk membentuk *dxk*-dimensional matrix W (dimana setiap kolom *k* yang terbentuk merepresentasikan eigen vector)

Menentukan jumlah *k* komponen utama (eigen vector) untuk mereduksi dimensi bisa dengan cara melihat kumulatif proporsi variance yang bisa dijelaskan setiap komponen utama

2.3.6.1 Menentukan k Komponen Utama dengan kumulatif proporsi variance

In [18]:

```
tot_eig_vals = np.sum(eigen_vals)
proporsi_variance = [(i/tot_eig_vals) for i in sorted(eigen_vals, reverse=True)]
kumulatif_proporsi_variance = np.cumsum(proporsi_variance)
```

In [19]:

```
kumulatif_proporsi_variance
```

Out[19]:

```
array([0.45536737, 0.85341224, 1.        ])
```

Analysis
Dengan k=2 (dua komponen utama pertama) sudah bisa menjelaskan +=80% variance (informasi) dari data dummy

2.3.6.2 Membentuk matrix W

Ambil dua eigen vector pertama dan susun secara horisontal untuk membentuk matrix **W**

In [20]:

```
PC_1 = eigen_pairs[0][1].reshape(3,1)
PC_2 = eigen_pairs[1][1].reshape(3,1)
matrix_w = np.hstack((PC_1,PC_2))
```

In [21]:

```
matrix_w
```

Out[21]:

```
array([[-0.37752765, -0.85952652],
       [-0.88332971,  0.44591347],
       [-0.27785158, -0.24975057]])
```

2.3.7 Gunakan matrix W untuk mentransformasi sampel dataset ke dalam ranah PCA yang baru

In [22]:

```
all_samples_transformed = matrix_w.T.dot(all_samples)
```

In [23]:

```
pd.DataFrame(all_samples_transformed)
```

Out[23]:

	0	1	2	3	4	5	6	7	8
0	1.026757	0.802664	0.453918	0.250617	-0.592331	-1.354991	0.224235	1.08049	-1.051787
1	-0.843672	0.237023	-1.438343	0.553137	0.094642	-0.840108	-2.434913	1.00171	-0.709822

2 rows × 40 columns

Analysis
Data di atas adalah sampel data dummy hasil transformasi ke dalam ranah PCA yang baru, dimana baris menyatakan jumlah PC terbentuk sedangkan kolom menyatakan jumlah sampel data

2.4 PCA Example Calculation for Breast Cancer Wisconsin Diagnostic Dataset

2.4.1 Siapkan data WDBC training

In [24]:

```
training_data = pickle.load(open("dataset/dataset.pkl", 'rb'))
training_data_prep = pickle.load(open("dataset/dataset_prep.pkl", 'rb'))
```

2.4.2 Load dataset

In [25]:

```
X_train = training_data['training']['data'][:,0:-1]
y_train = training_data['training']['data'][:,-1]
```

2.4.3 Standardisasi X_train

In [26]:

```
X_train_std = StandardScaler().fit_transform(X_train)
```

2.4.4 Hitung covariance matrix dari X_train

In [27]:

```
cov_mat_wdbc = np.cov(X_train_std.T)
```

In [28]:

```
pd.DataFrame(cov_mat_wdbc)
```

Out[28]:

	0	1	2	3	4	5	6	7	
0	1.002203	0.327077	1.000013	0.988914	0.189221	0.528061	0.677813	0.828805	0.14874
1	0.327077	1.002203	0.332531	0.319513	-0.011775	0.247362	0.292985	0.289665	0.08965
2	1.000013	0.332531	1.002203	0.988132	0.226338	0.578808	0.717621	0.857299	0.18609
3	0.988914	0.319513	0.988132	1.002203	0.200474	0.519326	0.687097	0.828914	0.15437
4	0.189221	-0.011775	0.226338	0.200474	1.002203	0.669680	0.541645	0.572018	0.56017
5	0.528061	0.247362	0.578808	0.519326	0.669680	1.002203	0.894756	0.847672	0.60949
6	0.677813	0.292985	0.717621	0.687097	0.541645	0.894756	1.002203	0.922626	0.51563
7	0.828805	0.289665	0.857299	0.828914	0.572018	0.847672	0.922626	1.002203	0.48031
8	0.148742	0.089651	0.186095	0.154379	0.560177	0.609494	0.515636	0.480319	1.00220
9	-0.295426	-0.068727	-0.244721	-0.265241	0.594070	0.561535	0.359191	0.183675	0.49809
10	0.680222	0.261637	0.693578	0.733495	0.324176	0.514673	0.632610	0.699123	0.31234
11	-0.074735	0.383410	-0.064139	-0.043922	0.027053	0.050479	0.084673	0.020156	0.13027
12	0.676066	0.267190	0.695638	0.728119	0.315983	0.565909	0.660385	0.710211	0.32768
13	0.731076	0.244089	0.740744	0.797458	0.268529	0.467206	0.615113	0.687347	0.22868
14	-0.186488	0.017559	-0.166532	-0.128560	0.330282	0.137656	0.120007	0.044766	0.21586
15	0.232639	0.199440	0.277482	0.237576	0.310008	0.735104	0.693145	0.507115	0.44181
16	0.186969	0.126368	0.220365	0.200314	0.242362	0.551537	0.685031	0.427545	0.34517
17	0.379518	0.169969	0.410410	0.374091	0.369223	0.641403	0.686076	0.611378	0.41747
18	-0.075886	0.010062	-0.051759	-0.046431	0.182830	0.241755	0.202290	0.121024	0.45295
19	-0.024962	0.058344	0.011081	-0.004142	0.264697	0.492626	0.470723	0.261836	0.34080
20	0.971391	0.351345	0.971467	0.962842	0.243805	0.560938	0.690294	0.839913	0.19580
21	0.304561	0.913679	0.310571	0.290756	0.060743	0.264831	0.297400	0.298933	0.11544
22	0.966564	0.357375	0.972142	0.958821	0.270528	0.616766	0.733337	0.866265	0.23247
23	0.942587	0.337090	0.943158	0.960288	0.238231	0.531571	0.676806	0.817544	0.18719
24	0.125187	0.076022	0.156313	0.133609	0.825910	0.570517	0.452996	0.459997	0.44099
25	0.418558	0.282960	0.461101	0.394708	0.493204	0.869271	0.761565	0.677851	0.48124
26	0.517074	0.290252	0.554544	0.502527	0.454732	0.822022	0.880752	0.748153	0.43576
27	0.742705	0.289271	0.769713	0.719362	0.532504	0.830032	0.859770	0.914259	0.44959
28	0.151966	0.114371	0.178429	0.131388	0.416057	0.513415	0.411288	0.385348	0.70786
29	0.010735	0.126948	0.054109	0.008749	0.510456	0.677475	0.522809	0.373698	0.44846

30 rows × 30 columns

2.4.5 Hitung eigen vector dan eigen value dari dekomposisi covariance matrix

In [29]:

```
eigen_vals, eigen_vecs = np.linalg.eig(cov_mat_wdbc)
```

In [30]:

```
eigen_vals.shape, eigen_vals
```

Out[30]:

```
((30,),
 array([1.34075237e+01, 5.57583121e+00, 2.88172583e+00, 1.98248759e+00,
        1.69040102e+00, 1.19918188e+00, 6.65811271e-01, 4.85268216e-01,
        3.86285667e-01, 3.50468676e-01, 3.03174298e-01, 2.72258795e-01,
        2.38314353e-01, 1.44161469e-01, 9.14330565e-02, 7.55558771e-02,
        5.93879845e-02, 5.24392723e-02, 4.73079416e-02, 1.24064291e-04,
        7.60153510e-04, 1.51324664e-03, 6.31364827e-03, 7.94084410e-03,
        1.53896919e-02, 1.59724926e-02, 2.39270335e-02, 2.35728449e-02,
        2.99319648e-02, 3.16151650e-02]))
```

In [31]:

```
eigen_vecs.shape, eigen_vecs
```

Out[31]:

```
((30, 30),
 array([[ 2.17057431e-01, -2.39775900e-01,  8.31060655e-03,
         -3.40193015e-02,  5.12734630e-02, -2.20945940e-02,
         -1.31638714e-01, -4.60976452e-02,  1.74010752e-01,
         -1.33805265e-01, -2.56398256e-02,  5.89029887e-02,
         -4.83977943e-02, -5.52553862e-02, -8.57275486e-02,
          1.43801349e-01, -2.00364354e-01,  9.70881634e-03,
         -2.80074209e-01,  6.98030280e-01, -2.23045237e-01,
          2.12174077e-01,  7.61235927e-02, -1.74390160e-01,
          9.70781690e-02, -1.46866692e-01, -8.78293674e-02,
         -4.35554320e-02, -9.13962385e-02, -1.47350002e-03],
        [ 1.01899728e-01, -6.15333635e-02, -4.26220397e-02,
          6.04567155e-01, -7.83891240e-02,  3.33296066e-02,
         -1.93240458e-02,  2.01141537e-01, -1.27047907e-01,
         -1.06163247e-01,  3.27072802e-01,  1.70171312e-01,
         -2.75407475e-01,  3.50419251e-02, -9.70344346e-02,
          1.41657823e-01,  5.78812085e-02, -5.69463836e-02,
         -3.92174212e-03, -3.51575646e-04,  1.00833953e-02,
          1.43059623e-03,  2.44219273e-02, -3.20696642e-02,
         -1.20938698e-01,  4.41118549e-02,  5.08790148e-02,
          4.85281560e-02,  7.58589671e-02,  5.14694971e-01],
        [ 2.25804376e-01, -2.20948876e-01,  8.68232367e-03,
         -3.48113744e-02,  4.92302321e-02, -2.14642056e-02,
         -1.20194615e-01, -5.55994395e-02,  1.79168555e-01,
         -1.24420963e-01,  1.26465772e-03,  3.39037595e-02,
         -7.08017340e-02, -4.65893243e-02, -6.77329016e-02,
          1.05109190e-01, -1.90436524e-01,  1.97666063e-02,
         -2.93438605e-01, -6.95659116e-01, -3.78689862e-01,
          8.26669162e-02,  6.82522377e-02, -1.50156420e-01,
          2.45421527e-02, -1.16563161e-01, -6.72562641e-02,
          7.53958263e-03, -5.69745753e-02, -2.76919670e-02],
        [ 2.19078681e-01, -2.36925147e-01, -2.67736696e-02,
         -4.97599792e-02,  1.92675117e-02,  5.85562253e-03,
         -6.33942531e-02, -9.57527245e-04,  1.58599160e-01,
         -1.18961755e-01, -1.02273555e-01,  5.08366112e-02,
         -1.09543286e-01,  1.30444985e-02, -2.80933461e-02,
          1.47125261e-01, -2.56622375e-01,  2.67214945e-01,
         -1.29388659e-01, -2.32994169e-02,  4.26246466e-01,
         -2.83850601e-01, -3.18060870e-01,  4.70099294e-01,
          1.33236198e-01,  1.79428701e-01, -3.95644070e-02,
          3.57222569e-02, -7.37310042e-02,  6.41668442e-02],
        [ 1.48588416e-01,  1.77062535e-01,  1.23746613e-01,
         -1.72319403e-01, -3.53763046e-01,  2.71336064e-01,
         -1.15506971e-01, -2.85401991e-01, -6.29310565e-02,
          1.17839523e-01,  1.47535514e-01,  2.53565728e-01,
         -1.67642124e-01, -4.28780946e-01, -1.49963822e-01,
          2.47695190e-01, -1.90326936e-01, -2.00884879e-01,
          3.21511607e-01, -6.36667116e-03,  2.92982613e-03,
          2.49028902e-03, -8.19754214e-02, -1.59080931e-02,
         -4.93837802e-02,  4.61321917e-02, -8.75031524e-02,
          3.12321730e-02, -1.51052153e-02, -8.87869086e-02],
        [ 2.41928271e-01,  1.39921191e-01,  7.23898200e-02,
         -2.69189891e-02,  5.87972064e-03,  9.35045991e-03,
          4.27691174e-02, -1.57806192e-01,  1.74614778e-01,
         -2.90580680e-02,  3.27007129e-01, -1.52878711e-01,
```

```
        -1.39926313e-01, -2.25357185e-02,  2.15569158e-01,
        -2.08366068e-01,  2.28623479e-03, -1.47232452e-01,
        -2.41805812e-01,  4.95303886e-02,  5.02077932e-02,
        -1.67452479e-02, -1.11303290e-02,  2.99281925e-01,
        -2.87215954e-01, -3.19303484e-01,  1.10297390e-01,
        -5.34219019e-02,  4.72184742e-01, -1.41186465e-01],
       [ 2.57761371e-01,  6.00889218e-02, -6.24351727e-03,
        -1.87723504e-02,  8.83143741e-02,  1.19821107e-02,
        -8.06508710e-02, -1.00743710e-01, -1.16735192e-02,
         1.44987657e-01, -9.94907550e-02, -1.24298416e-01,
        -3.77907967e-01,  1.82282980e-01, -1.22411439e-01,
        -2.85104543e-01,  4.79553478e-02, -1.95019186e-03,
         3.71197778e-02,  2.07981749e-02,  5.29929685e-03,
         3.12793023e-02, -5.04078616e-01, -4.08132351e-01,
         4.71685163e-02,  1.04856866e-01,  3.40004498e-01,
         1.68245649e-01, -2.21024918e-02, -5.23217734e-02],
       [ 2.60382814e-01, -4.05327434e-02,  3.31201149e-02,
        -6.65966837e-02, -3.70162729e-02,  3.09958493e-02,
        -1.38947658e-01, -1.83449390e-01,  7.36311590e-02,
        -2.22997644e-02,  9.66513835e-02, -1.07919309e-02,
        -1.28309296e-01,  2.17074665e-01, -2.01147132e-01,
        -3.78205963e-01,  2.31849917e-02, -8.30749423e-04,
         2.14406541e-01, -2.89135472e-04,  4.87900870e-03,
         2.03297229e-02,  5.69221648e-01,  2.39957929e-01,
         1.17808100e-01,  1.76885235e-01,  2.05229432e-01,
        -1.33882213e-01, -2.65993212e-01,  5.26459366e-02],
       [ 1.42086256e-01,  1.93611573e-01,  3.15264736e-02,
        -6.02587961e-02, -2.89882834e-01, -3.53413668e-01,
        -8.88439538e-02, -1.52937047e-01, -4.15289978e-01,
        -5.21363900e-01, -4.73242467e-02, -3.39567619e-01,
        -1.05418729e-01, -1.45989569e-02, -6.23928967e-02,
         1.82643069e-01,  2.06045381e-01,  1.69444644e-01,
        -5.26269328e-02, -1.36417500e-04,  6.67891042e-03,
         8.53041956e-05,  6.47817945e-03, -1.98693172e-02,
         1.02088937e-02,  1.35082575e-02, -5.69303930e-02,
        -5.94373109e-02, -1.26206337e-02, -7.37448460e-02],
       [ 7.15926428e-02,  3.66647802e-01,  2.47543702e-02,
        -4.77276023e-02, -5.79845971e-02,  1.24979276e-01,
         2.94944563e-01, -1.71084850e-01,  8.54173356e-02,
        -1.16367275e-01,  7.23313725e-02,  2.21445701e-01,
        -1.75057295e-01,  4.08740565e-01,  4.91316971e-01,
         6.40295407e-02, -1.09650520e-01,  7.94925741e-02,
        -3.09469192e-02, -4.48413418e-03, -8.07980953e-03,
         1.15339549e-03,  4.80386040e-02, -1.28238539e-01,
         2.53928054e-01, -1.72240018e-03, -2.57039830e-01,
         8.88308055e-02, -1.39970882e-01,  8.84367350e-02],
       [ 2.04691204e-01, -1.10269914e-01, -2.61600856e-01,
        -1.01823291e-01, -1.66747427e-01,  3.83404235e-02,
         3.12163811e-01,  9.01640280e-02, -2.06300980e-01,
         9.08262963e-02,  2.09705897e-02,  2.08147970e-03,
         7.90976290e-02, -2.35310557e-02, -1.01213546e-01,
        -7.53219059e-02,  1.49635865e-01, -3.17142219e-01,
        -4.00528315e-02, -9.47190762e-03, -1.15520840e-01,
        -2.03167762e-01, -1.88025040e-01,  1.21931320e-01,
         4.21289481e-01, -3.80467186e-01,  3.25137325e-02,
        -2.68478540e-01, -1.26784544e-01,  1.31712048e-01],
       [ 2.02094854e-02,  7.58423303e-02, -3.65249385e-01,
         3.65445327e-01, -1.98434626e-01,  3.31816944e-02,
        -3.54972839e-03, -5.63949664e-01,  2.20535472e-01,
         1.17462584e-01, -3.49045609e-01, -2.27290161e-01,
         2.62826366e-01, -1.45537274e-02,  1.38585100e-02,
```

```
        3.69765832e-02, -6.36669493e-02, -2.15109059e-02,
       -1.33277984e-02, -1.45437156e-03,  8.01148689e-03,
       -5.13246965e-03,  1.71721211e-02, -1.86586201e-02,
       -5.66156004e-02, -5.17195619e-03,  2.93018454e-02,
        1.41488638e-02,  5.93401875e-02,  2.19853374e-01],
      [ 2.10070529e-01, -9.44153634e-02, -2.60287136e-01,
       -9.17718274e-02, -1.34976345e-01,  4.00389749e-03,
        3.28514948e-01,  6.14012646e-02, -1.62763810e-01,
        1.48869353e-01,  1.54391341e-01, -8.97142744e-02,
        9.77981905e-02,  2.10345610e-02,  1.38791297e-02,
       -2.27491530e-02,  1.33446002e-02, -1.80647124e-01,
       -2.83427505e-01,  1.36716149e-02,  3.34198370e-03,
        2.69610787e-01, -2.76802763e-02,  1.01824944e-01,
       -3.47707487e-01,  4.28012505e-01, -1.35108848e-01,
        2.10941475e-01, -2.66986119e-01, -1.19618410e-01],
      [ 1.99926314e-01, -1.57130070e-01, -2.12088203e-01,
       -1.15348752e-01, -1.36352863e-01,  6.30015249e-02,
        3.44836655e-01,  1.57711763e-01, -1.86190380e-01,
        1.12299274e-01, -6.53225509e-02, -4.44715154e-02,
       -4.48317155e-02, -2.73168982e-02, -7.12952170e-02,
       -5.20269871e-02, -3.14170471e-01,  4.68282611e-01,
        1.92401419e-01,  1.37658750e-03,  8.63320795e-02,
       -3.93736494e-02,  2.67199776e-01, -2.43617790e-01,
       -7.54542225e-02, -1.18860558e-01,  1.66651670e-02,
        3.64795274e-02,  3.61075092e-01, -8.48416316e-03],
      [ 2.31645996e-02,  1.90534593e-01, -2.96997093e-01,
       -5.15589478e-02, -2.63105212e-01,  3.50438025e-01,
       -3.44866742e-01,  5.03384337e-01,  2.16149081e-01,
       -1.73086759e-01, -9.47328983e-02, -3.00068384e-01,
       -2.16600847e-02,  2.16764167e-01,  2.95041912e-03,
        3.42339383e-02, -1.25609238e-01, -1.94366068e-01,
        1.01374672e-01, -1.43298803e-03, -1.21313624e-03,
        9.80183639e-03,  2.17607164e-03, -5.68051443e-03,
       -2.50074146e-02, -2.56397836e-03, -1.00038880e-01,
       -1.55126919e-02,  4.14031670e-02, -8.38996097e-02],
      [ 1.75338940e-01,  2.24035281e-01, -1.69104563e-01,
        3.83986522e-02,  2.68271420e-01, -5.33599238e-02,
        9.57670698e-03,  1.29184766e-01,  1.52318574e-01,
       -7.64503807e-02,  2.07561813e-01, -2.28023381e-01,
        7.44347798e-02, -4.83874175e-01,  2.32408617e-01,
       -1.05048625e-01, -3.03814596e-02,  2.38248181e-01,
        2.16448080e-01,  2.46175350e-03, -3.02297199e-03,
       -7.04221395e-03, -4.67771761e-02, -3.17531588e-02,
       -6.62735974e-02, -1.84716203e-01,  4.46412298e-02,
        7.85751754e-02, -4.45795368e-01,  1.23861393e-01],
      [ 1.50796174e-01,  2.02018669e-01, -1.95783187e-01,
        8.92428448e-03,  3.47644795e-01, -2.65657267e-02,
       -1.82054829e-01,  5.08218193e-02, -3.21891646e-01,
        2.31861642e-01, -3.45778083e-01,  1.07130784e-01,
       -2.74332145e-01, -8.58607967e-02,  2.30837513e-01,
        1.27500527e-01, -9.85604840e-02, -8.69697233e-02,
       -1.10116082e-01, -8.64384372e-03, -1.43512667e-02,
       -1.72594857e-03,  1.17243657e-01,  6.60244137e-02,
       -1.47877551e-01,  1.12669257e-01, -1.24618823e-02,
       -4.66037682e-01,  1.67261108e-02,  1.34911218e-02],
      [ 1.84482732e-01,  1.33312587e-01, -2.28363359e-01,
       -5.03967547e-02,  1.88824258e-01,  9.63812299e-03,
       -3.52366651e-01, -7.24635948e-02, -2.96494698e-01,
        3.80429874e-02,  3.23570555e-01,  1.84383111e-01,
        4.89596458e-01,  2.01177984e-01,  2.67374122e-02,
        1.86855545e-01, -1.09571048e-01,  3.89915557e-02,
```

```
        -4.07884600e-02, -1.94020905e-03,  2.36529970e-02,
        -3.07574940e-02,  1.29210498e-05, -2.49173666e-02,
         1.65812176e-01, -6.46653576e-04,  2.74745216e-01,
         2.10879588e-01,  1.16788450e-01, -7.79498239e-02],
       [ 5.03977614e-02,  1.74132183e-01, -2.95181277e-01,
        -5.22837286e-02, -2.41214492e-01, -4.99505362e-01,
        -7.73023944e-02,  1.30988449e-01,  4.03819955e-01,
         2.86419042e-01,  1.10631603e-01,  2.96642953e-01,
        -1.01070322e-01,  3.55144573e-02, -1.03998061e-01,
         1.47103733e-01,  2.94961595e-01,  1.92960795e-01,
        -3.36964518e-03,  8.96026955e-04,  3.64766827e-03,
        -9.22986419e-03,  1.14239696e-02, -2.47636778e-02,
         2.90886325e-03,  8.29848344e-03, -1.50534266e-02,
        -1.00784141e-01,  2.89380768e-02, -1.44170023e-01],
       [ 1.06999631e-01,  2.76203478e-01, -2.24180066e-01,
         6.08042990e-03,  2.64045571e-01,  8.03000157e-02,
         1.31582855e-01,  2.08477668e-02,  6.20346412e-02,
        -4.12381419e-01, -2.20030242e-01,  3.68032248e-01,
         5.32601625e-02, -2.09005637e-01, -3.26991398e-01,
        -2.94562445e-01,  1.40944824e-01, -7.39520068e-02,
        -5.58354885e-02,  1.10732761e-03,  6.77981155e-03,
         2.46738109e-02,  1.29382781e-02,  3.31412546e-02,
         5.91879407e-02,  1.09967439e-01, -1.91218834e-01,
         1.81763260e-01,  2.32348095e-01, -2.27186938e-02],
       [ 2.26607291e-01, -2.23083155e-01,  5.17653925e-02,
        -1.43404797e-02,  5.30233342e-03, -3.81390988e-03,
        -2.85517841e-02,  1.54659935e-02,  7.66532031e-02,
        -1.15173672e-01, -1.13234362e-01,  1.06069133e-01,
         1.02287637e-01, -3.02207010e-02,  1.75595287e-01,
         8.91532530e-02,  2.19656351e-01, -2.14616949e-01,
         9.12171768e-02, -1.30998208e-01,  6.38000035e-01,
         4.08680324e-01,  8.97224067e-02, -2.46386958e-01,
         2.31182058e-02, -1.73041971e-01,  3.37385369e-02,
        -8.34458400e-02, -1.25218964e-02, -1.34992298e-02],
       [ 1.04175660e-01, -4.70704638e-02,  6.50056537e-02,
         6.25307379e-01, -1.20840762e-01,  5.16595612e-02,
        -5.55280228e-03,  5.91333866e-02, -1.14439645e-01,
         2.32869311e-02, -2.39826489e-02,  9.93480226e-02,
         3.41056715e-02, -4.75880318e-02,  8.86516976e-02,
        -1.72516834e-01, -7.09357817e-02,  9.75741637e-02,
         3.03966860e-02,  1.77441736e-03, -1.65416371e-02,
        -4.71897045e-04, -3.13233088e-02,  4.89112703e-02,
         1.56234508e-01, -4.53446043e-02, -8.92358010e-02,
        -6.19864560e-02, -1.12264524e-01, -6.60124232e-01],
       [ 2.35493815e-01, -2.02364829e-01,  5.19203897e-02,
        -1.18213495e-02,  1.55763466e-02, -1.37437420e-02,
        -1.40881128e-02,  6.07854867e-03,  8.62407957e-02,
        -8.79518877e-02, -6.04411619e-02,  4.13547795e-02,
         8.35671938e-02, -2.93816214e-02,  2.07157318e-01,
         5.58938900e-02,  2.11237066e-01, -2.13060076e-01,
         3.72967213e-02,  7.90679497e-02, -3.42364561e-02,
        -7.20841569e-01,  1.74338621e-01, -2.98693465e-01,
        -2.13330056e-01,  1.48164462e-01, -3.42272114e-02,
         1.01387588e-01, -1.72037363e-02, -9.16682872e-02],
       [ 2.23108037e-01, -2.23733767e-01,  1.84630890e-02,
        -2.85541683e-02, -2.06669089e-02,  2.88599058e-02,
         4.54833924e-02,  5.75679238e-02,  4.64077022e-02,
        -1.17130169e-01, -2.01565377e-01,  1.02835067e-01,
         6.21390860e-02,  2.93636106e-02,  3.30592438e-01,
         1.22930065e-01,  2.40228068e-01,  8.39758736e-02,
         4.18019107e-01,  3.56094006e-02, -4.36295805e-01,
```

```
         2.42237975e-01, -1.66040338e-01,  2.99889619e-01,
        -1.71635006e-02,  1.36159010e-01,  1.69387827e-01,
         6.55440001e-02,  1.89626924e-01,  3.28192704e-02],
       [ 1.29967792e-01,  1.68981821e-01,  2.73405486e-01,
        -4.61061305e-02, -3.21436423e-01,  3.41605847e-01,
        -1.33500096e-01,  1.51450227e-01, -5.03716738e-02,
         1.62909944e-01, -1.85264461e-01,  1.13402272e-01,
         1.62635187e-01, -1.85067218e-01,  9.67831998e-02,
        -1.39325697e-01,  2.92374516e-01,  3.28085669e-01,
        -4.36312990e-01,  5.46238825e-03, -9.88558908e-03,
        -1.52330033e-03,  5.88593828e-02, -7.73072221e-03,
         2.73381351e-02,  1.50291103e-02,  1.63735973e-01,
         2.48214829e-02, -4.34669340e-02,  1.83653961e-01],
       [ 2.10301491e-01,  1.42217285e-01,  2.32022838e-01,
         8.69426198e-02,  1.21793645e-01, -3.96556736e-02,
         1.59189906e-01,  9.59429316e-02,  1.86198469e-01,
         1.13327973e-01,  1.66522718e-01, -3.32162831e-01,
         1.55231587e-01, -1.63116370e-01, -4.27779114e-02,
         1.28190383e-01, -1.80212753e-02, -7.09397237e-02,
        -1.24054240e-02, -1.49298183e-02, -1.76289250e-03,
         4.38446813e-02,  2.69919765e-02, -1.22743294e-01,
         4.19513106e-01,  4.69324047e-01, -6.81833934e-02,
        -2.41072458e-01,  2.91336997e-01,  3.61549869e-02],
       [ 2.26321939e-01,  1.04546705e-01,  1.66840008e-01,
         7.07202788e-02,  2.01612756e-01, -1.95439989e-02,
        -2.11308005e-02,  5.84189198e-02, -6.05801979e-02,
         3.45412215e-01, -1.98153988e-01, -2.19446107e-01,
        -1.34030575e-01,  8.61457737e-02, -1.77704918e-01,
         2.49406179e-01,  1.85955731e-01, -5.67367381e-02,
         4.94681945e-02,  2.67192701e-03,  1.34345360e-02,
         3.99173541e-03,  1.88982097e-01,  1.98104932e-01,
         1.03032235e-01, -2.65433310e-01, -2.64175663e-01,
         5.13802793e-01, -1.75026419e-03, -1.69957438e-02],
       [ 2.49366974e-01, -6.52474265e-03,  1.73776407e-01,
        -1.22778229e-02,  5.05917893e-02,  7.39659422e-03,
        -1.65254011e-01, -5.44867900e-02, -5.14550563e-02,
         1.00786683e-01,  8.97380342e-02,  2.89926494e-02,
         2.89467581e-01,  2.51902041e-01, -1.49865986e-01,
        -2.11058112e-01,  5.37554107e-02,  1.66214504e-01,
         1.79949032e-01, -2.86651435e-03, -3.47163852e-02,
         8.15962895e-03, -2.61519141e-01, -5.54483474e-02,
        -2.76903308e-01, -7.58740957e-02, -5.57112477e-01,
        -2.91255816e-01, -9.74019717e-03,  1.58761692e-01],
       [ 1.22964478e-01,  1.50772459e-01,  2.58227342e-01,
         2.31372803e-02, -2.27207165e-01, -5.12424044e-01,
        -3.39657547e-02,  2.20651883e-01, -3.74726818e-02,
         2.66734992e-02, -2.35678489e-01,  1.22131956e-01,
         1.74817390e-01, -3.28631090e-02,  1.09678925e-01,
        -2.45857619e-01, -4.52162352e-01, -2.77252434e-01,
         3.52058630e-02, -3.73618466e-04, -1.19769470e-02,
         8.20668943e-03, -8.66412936e-03,  3.68097576e-02,
        -1.36829881e-02, -1.90669051e-02,  6.94662334e-02,
         1.72300827e-01,  3.53830067e-03,  1.86866161e-01],
       [ 1.33513179e-01,  2.77470544e-01,  2.27018697e-01,
         7.63374541e-02,  9.37933131e-02,  9.28440509e-02,
         3.58977254e-01,  8.29711167e-02,  1.49475604e-01,
        -1.21790053e-01, -1.28659531e-01,  5.56977457e-02,
         1.72653463e-01,  2.18646738e-01, -3.13896769e-01,
         3.44924287e-01, -8.29757745e-02,  9.71836113e-03,
         2.52489145e-02,  1.27399238e-03, -1.35322627e-02,
        -3.43062875e-02, -3.27019772e-02,  2.20941391e-02,
```

```
          -3.03045921e-01, -1.16213023e-01,  3.69406858e-01,
          -2.02306034e-01, -1.91564969e-01, -1.12861619e-01]]))
```

2.4.6 Urutkan eigen vector dengan mengurutkan nilai eigen value secara decreasing

In [32]:

```
# buat list yang isinya pasangan eigen value dengan masing-masing eigen vector
eigen_pairs = [
    (np.abs(eigen_vals[i]), eigen_vecs[:,i]) for i in range(len(eigen_vals))
]

# cetak eigen pairs sebelum diurutkan
eigen_pairs
```

```
  array([-0.13163871, -0.01932405, -0.12019461, -0.06339425, -0.11550697,
          0.04276912, -0.08065087, -0.13894766, -0.08884395,  0.29494456,
          0.31216381, -0.00354973,  0.32851495,  0.34483666, -0.34486674,
          0.00957671, -0.18205483, -0.35236665, -0.07730239,  0.13158286,
         -0.02855178, -0.0055528 , -0.01408811,  0.04548339, -0.1335001 ,
          0.15918991, -0.0211308 , -0.16525401, -0.03396575,  0.3589772
5])),
 (0.48526821579616164,
  array([-0.04609765,  0.20114154, -0.05559944, -0.00095753, -0.28540199,
         -0.15780619, -0.10074371, -0.18344939, -0.15293705, -0.17108485,
          0.09016403, -0.56394966,  0.06140126,  0.15771176,  0.50338434,
          0.12918477,  0.05082182, -0.07246359,  0.13098845,  0.02084777,
          0.01546599,  0.05913339,  0.00607855,  0.05756792,  0.15145023,
          0.09594293,  0.05841892, -0.05448679,  0.22065188,  0.0829711
2])),
 (0.38628566692728705,
  array([ 0.17401075, -0.12704791,  0.17916855,  0.15859916, -0.06293106,
          0.17461478, -0.01167352,  0.07363116, -0.41528998,  0.08541734,
         -0.20630098,  0.22053547, -0.16276381, -0.18619038,  0.21614908,
          0.15231857, -0.32189165, -0.2964947 ,  0.40381996,  0.06203464,
```

In [33]:

```
# urutkan tuples eigen_pairs secara descending order berdasarkan eigen value
eigen_pairs.sort(key=lambda x:x[0], reverse=True)

# cetak eigen pairs setelah diurutkan
eigen_pairs
```

```
          0.26827142,  0.3476448 ,  0.18882426, -0.24121449,  0.26404557,
          0.00530233, -0.12084076,  0.01557635, -0.02066691, -0.32143642,
          0.12179365,  0.20161276,  0.05059179, -0.22720717,  0.0937933
1])),
 (1.1991818817408058,
  array([-0.02209459,  0.03332961, -0.02146421,  0.00585562,  0.27133606,
          0.00935046,  0.01198211,  0.03099585, -0.35341367,  0.12497928,
          0.03834042,  0.03318169,  0.0040039 ,  0.06300152,  0.35043803,
         -0.05335992, -0.02656573,  0.00963812, -0.49950536,  0.08030002,
         -0.00381391,  0.05165956, -0.01374374,  0.02885991,  0.34160585,
         -0.03965567, -0.019544  ,  0.00739659, -0.51242404,  0.0928440
5])),
 (0.6658112712454161,
  array([-0.13163871, -0.01932405, -0.12019461, -0.06339425, -0.11550697,
          0.04276912, -0.08065087, -0.13894766, -0.08884395,  0.29494456,
          0.31216381, -0.00354973,  0.32851495,  0.34483666, -0.34486674,

          0.00957671, -0.18205483, -0.35236665, -0.07730239,  0.13158286,
         -0.02855178, -0.0055528 , -0.01408811,  0.04548339, -0.1335001 ,
          0.15918991, -0.0211308 , -0.16525401, -0.03396575,  0.3589772
```

2.4.7 Pilih k eigen vector dengan beberapa eigen value terbesar pertama untuk membentuk *dxk*-dimensional matrix W (dimana setiap kolom *k* yang terbentuk merepresentasikan eigen vector)

2.4.7.1 Menentukan k komponen utama

In [34]:

```
tot_eig_vals = np.sum(eigen_vals)
proporsi_variance = [(i/tot_eig_vals) for i in sorted(eigen_vals, reverse=True)]
kumulatif_proporsi_variance = np.cumsum(proporsi_variance)
```

In [35]:

```
kumulatif_proporsi_variance
```

Out[35]:

```
array([0.44593522, 0.63138778, 0.72723419, 0.79317187, 0.84939473,
       0.88927961, 0.91142454, 0.9275646 , 0.94041249, 0.9520691 ,
       0.9621527 , 0.97120805, 0.9791344 , 0.98392922, 0.98697029,
       0.98948329, 0.99145854, 0.99320267, 0.99477613, 0.99582766,
       0.9968232 , 0.99761901, 0.99840305, 0.99893429, 0.99944615,
       0.99971027, 0.99992026, 0.99997059, 0.99999587, 1.        ])
```

Analysis

Dengan k=5 (lima komponen utama pertama) dapat menjelaskan variance pada data latih >=80%

2.4.7.2 Membentuk matrix W

Ambil lima eigen vector pertama dan susun secara horisontal

In [36]:

```
PC_1 = eigen_pairs[0][1].reshape(30,1)
PC_2 = eigen_pairs[1][1].reshape(30,1)
PC_3 = eigen_pairs[2][1].reshape(30,1)
PC_4 = eigen_pairs[3][1].reshape(30,1)
PC_5 = eigen_pairs[4][1].reshape(30,1)
matrix_w = np.hstack((PC_1,PC_2, PC_3, PC_4, PC_5))
```

In [37]:

```
df_matrix_w = pd.DataFrame(matrix_w, columns=['PC-'+str(i+1) for i in range(5)])
```

In [38]:

```
df_matrix_w
```

Out[38]:

	PC-1	PC-2	PC-3	PC-4	PC-5
0	0.217057	-0.239776	0.008311	-0.034019	0.051273
1	0.101900	-0.061533	-0.042622	0.604567	-0.078389
2	0.225804	-0.220949	0.008682	-0.034811	0.049230
3	0.219079	-0.236925	-0.026774	-0.049760	0.019268
4	0.148588	0.177063	0.123747	-0.172319	-0.353763
5	0.241928	0.139921	0.072390	-0.026919	0.005880
6	0.257761	0.060089	-0.006244	-0.018772	0.088314
7	0.260383	-0.040533	0.033120	-0.066597	-0.037016
8	0.142086	0.193612	0.031526	-0.060259	-0.289883
9	0.071593	0.366648	0.024754	-0.047728	-0.057985
10	0.204691	-0.110270	-0.261601	-0.101823	-0.166747
11	0.020209	0.075842	-0.365249	0.365445	-0.198435
12	0.210071	-0.094415	-0.260287	-0.091772	-0.134976
13	0.199926	-0.157130	-0.212088	-0.115349	-0.136353
14	0.023165	0.190535	-0.296997	-0.051559	-0.263105
15	0.175339	0.224035	-0.169105	0.038399	0.268271
16	0.150796	0.202019	-0.195783	0.008924	0.347645
17	0.184483	0.133313	-0.228363	-0.050397	0.188824
18	0.050398	0.174132	-0.295181	-0.052284	-0.241214
19	0.107000	0.276203	-0.224180	0.006080	0.264046
20	0.226607	-0.223083	0.051765	-0.014340	0.005302
21	0.104176	-0.047070	0.065006	0.625307	-0.120841
22	0.235494	-0.202365	0.051920	-0.011821	0.015576
23	0.223108	-0.223734	0.018463	-0.028554	-0.020667
24	0.129968	0.168982	0.273405	-0.046106	-0.321436
25	0.210301	0.142217	0.232023	0.086943	0.121794
26	0.226322	0.104547	0.166840	0.070720	0.201613
27	0.249367	-0.006525	0.173776	-0.012278	0.050592
28	0.122964	0.150772	0.258227	0.023137	-0.227207
29	0.133513	0.277471	0.227019	0.076337	0.093793

2.4.8 Gunakan matrix W untuk mentransformasi sampel dataset ke dalam ranah PCA yang baru

In [39]:

```
X_train_std_transformed = X_train_std.dot(matrix_w)
```

In [40]:

```
pd.DataFrame(X_train_std_transformed)
```

Out[40]:

	0	1	2	3	4
0	-0.875937	-2.571309	0.548511	1.635793	0.262836
1	-2.505128	0.192453	0.404965	-2.447875	-0.600618
2	-1.354117	0.417851	-2.545009	1.218872	1.766641
3	4.854091	3.017576	1.626587	0.702197	2.814424
4	2.926225	-1.866546	-2.486468	-0.104045	-0.181489
...	...	...	...	...	...
450	-3.730069	-1.364877	-1.713709	0.555606	0.051602
451	-1.894651	0.942551	-0.074451	0.458228	-1.348708
452	-3.180442	-1.992055	-0.009280	1.811156	-0.312800
453	-0.568062	-0.906734	0.314257	0.535105	1.395945
454	3.171861	-1.573469	-1.196476	0.677494	-0.917171

455 rows × 5 columns