20230510_Wirtschaftsinformatik_MV2

May 10, 2023

[1]:
```
# kovarianzmatrix
```

[2]:
```
DLZ = [84, 82, 81, 89, 73, 94, 92, 70, 77, 95]
Output = [85, 82, 72, 77, 75, 89, 95, 84, 77, 94]
Q = [97, 94, 93, 95, 88, 82, 78, 84, 69, 78]
```

[3]:
```
import numpy as np
```

[4]:
```
data = np.array([DLZ, Output, Q])
```

[5]:
```
data
```

[5]:
```
array([[84, 82, 81, 89, 73, 94, 92, 70, 77, 95],
       [85, 82, 72, 77, 75, 89, 95, 84, 77, 94],
       [97, 94, 93, 95, 88, 82, 78, 84, 69, 78]])
```

[6]:
```
# kovarianzmatrix

np.cov(data, bias=True)
```

[6]:
```
array([[ 68.81,  39.8 ,  -5.96],
       [ 39.8 ,  56.4 , -24.1 ],
       [ -5.96, -24.1 ,  75.56]])
```

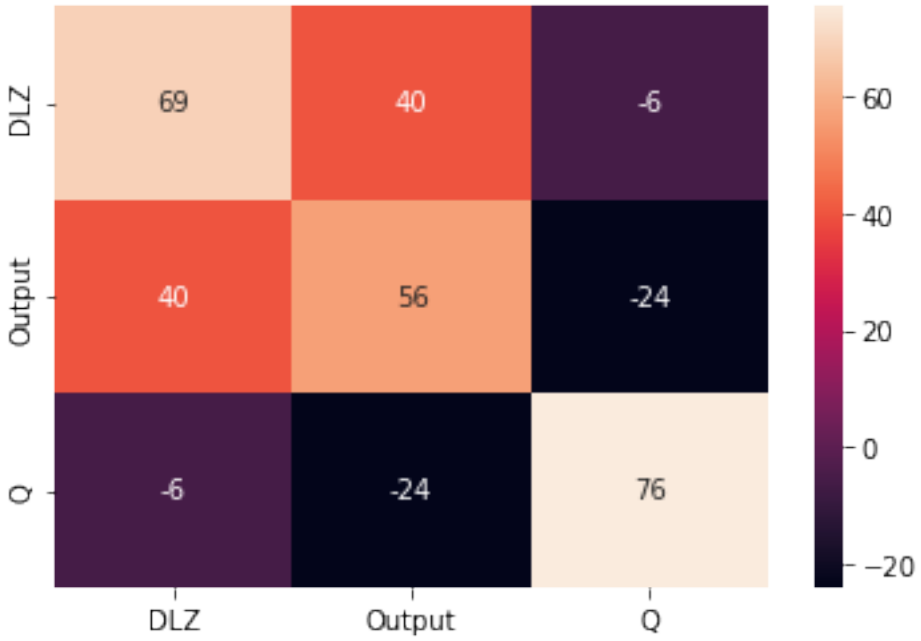
[7]:
```
cov_matrix = np.cov(data, bias=True)
```

[8]:
```
# graphische darstellung
```

[12]:
```
import seaborn as sns
import matplotlib.pyplot as plt

labels = ['DLZ', 'Output', 'Q']

sns.heatmap(cov_matrix, xticklabels=labels, yticklabels=labels, annot=True)
plt.show()
```



[13]:
```
# eigenwerte und eigenvektoren der kovarianzmatrix

eigwerte, eigenvektoren = np.linalg.eig(cov_matrix)
```

[14]:
```
eigwerte
```

[14]:
```
array([ 18.34887097, 114.12799976,  68.29312927])
```

[15]:
```
eigenvektoren
```

[15]:
```
array([[ 0.57763966, -0.61225795,  0.53988206],
       [-0.77206188, -0.62453512,  0.11779789],
       [-0.26505261,  0.48486709,  0.83345727]])
```

[16]:
```
# umsetzung hauptkomponenten Bildkompression
```

[19]:
```
from matplotlib.image import imread

image_raw = imread('/Users/h4/Desktop/20230510_granada.jpeg')
plt.figure(figsize=[12,8])
plt.imshow(image_raw)
plt.show()
```

[21]:
```
#nicht prüfungsrelevant

image_sum = image_raw.sum(axis=2)
image_bw = image_sum/image_sum.max()

plt.figure(figsize=[12,8])

plt.imshow(image_bw,cmap=plt.cm.gray)
plt.show()
```

[26]:
```
np.shape(image_bw)
```

[26]:
```
(497, 746)
```

[24]:
```
# hauptkomponenten vom Bild

from sklearn.decomposition import PCA, IncrementalPCA

pca = PCA() # Initialisierung vom PCA
pca.fit(image_bw) # hier erfolgt die PCA Analyse
```

[24]:
```
PCA()
```

[30]:
```
# BildKompression mittels PCA

i_pca = IncrementalPCA(n_components=40) # nur mit 40 Komponenten, anstatt 497,␣
 ↪746 Dimensionen
image_compression = i_pca.inverse_transform(i_pca.fit_transform(image_bw))
# hier erzeugen wir das Bild aus den Hauptkomponenten!!!

plt.figure(figsize=[12,8])
plt.imshow(image_compression, cmap=plt.cm.gray)
plt.show()
```

[ ]:
```
# Übung für die Prüfung.
# Bitte ermittle die erste 2 Hauptkomponenten der dataset "data" oben.
# Bitte stelle die Daten in den neuen 2 Dimensionalen Raum dar. (#neu)
```