k-means

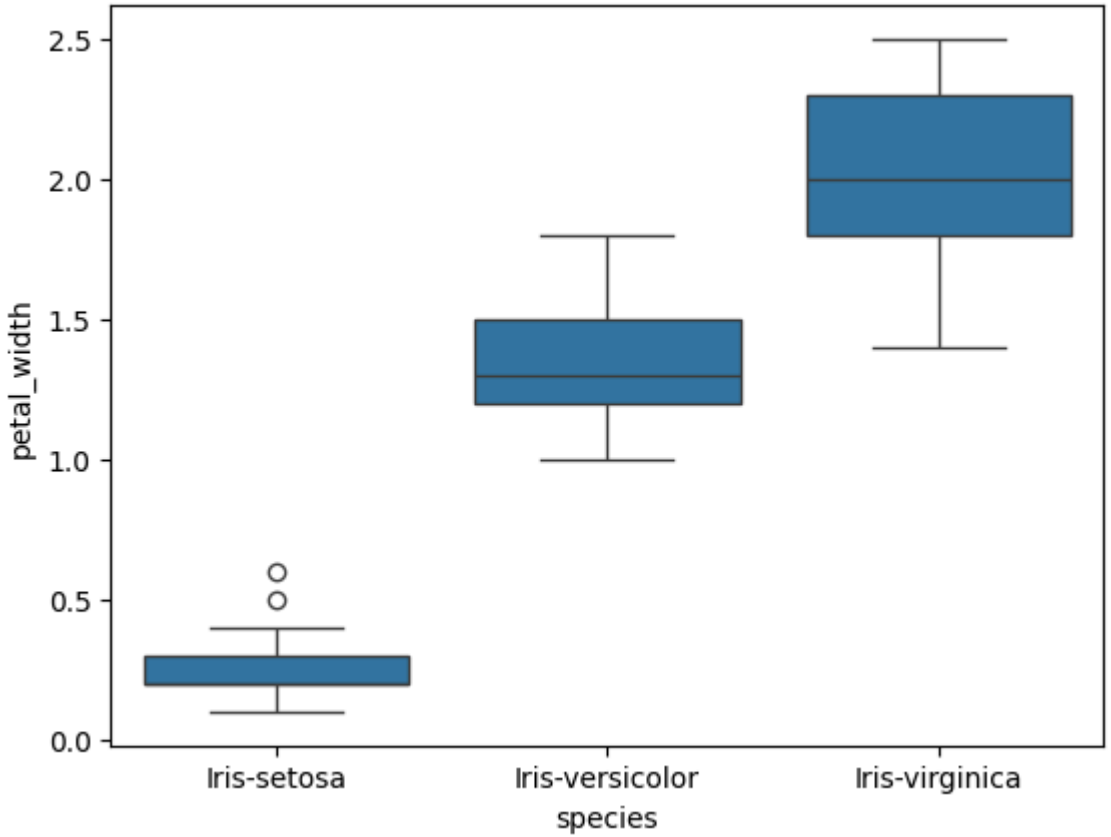
In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
```

In [ ]:
```
iris = pd.read_csv("IRIS.csv")
```

In [ ]:
```
iris.info()
```

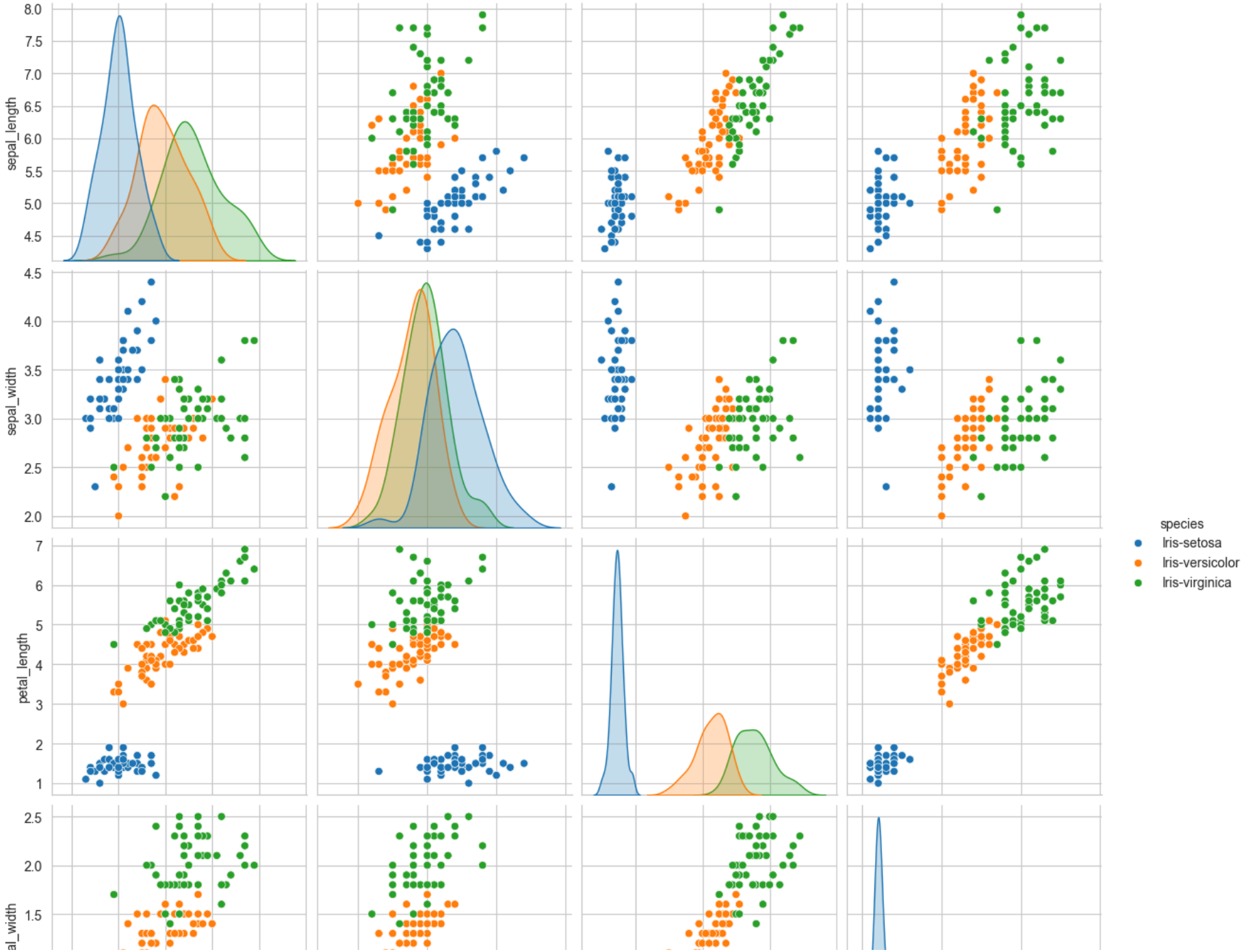
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

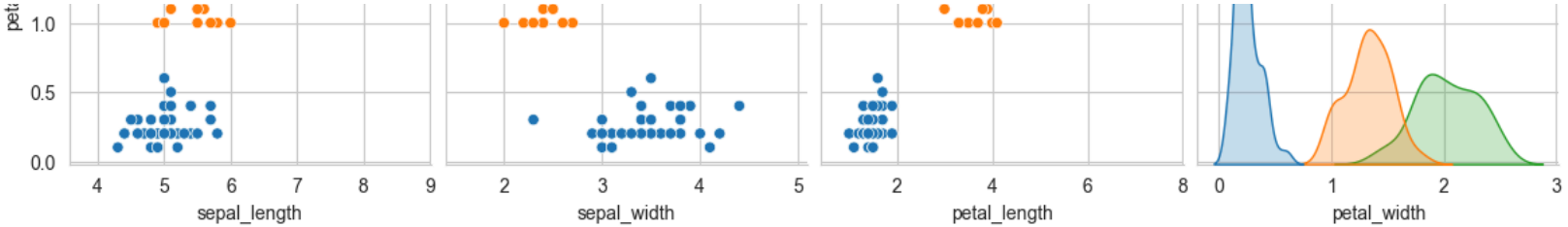
In [ ]:
```
sns.boxplot(x = "species",y = "petal_width" , data = iris)
plt.show()
```



In [ ]:
```
sns.set_style("whitegrid")
sns.pairplot(iris, hue = "species" , size=3)
plt.show()
```

```
c:\Users\Deepthi\AppData\Local\Programs\Python\Python38\lib\site-packages\seaborn\axisgrid.py:2100: UserWarning: The `size` par
ameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

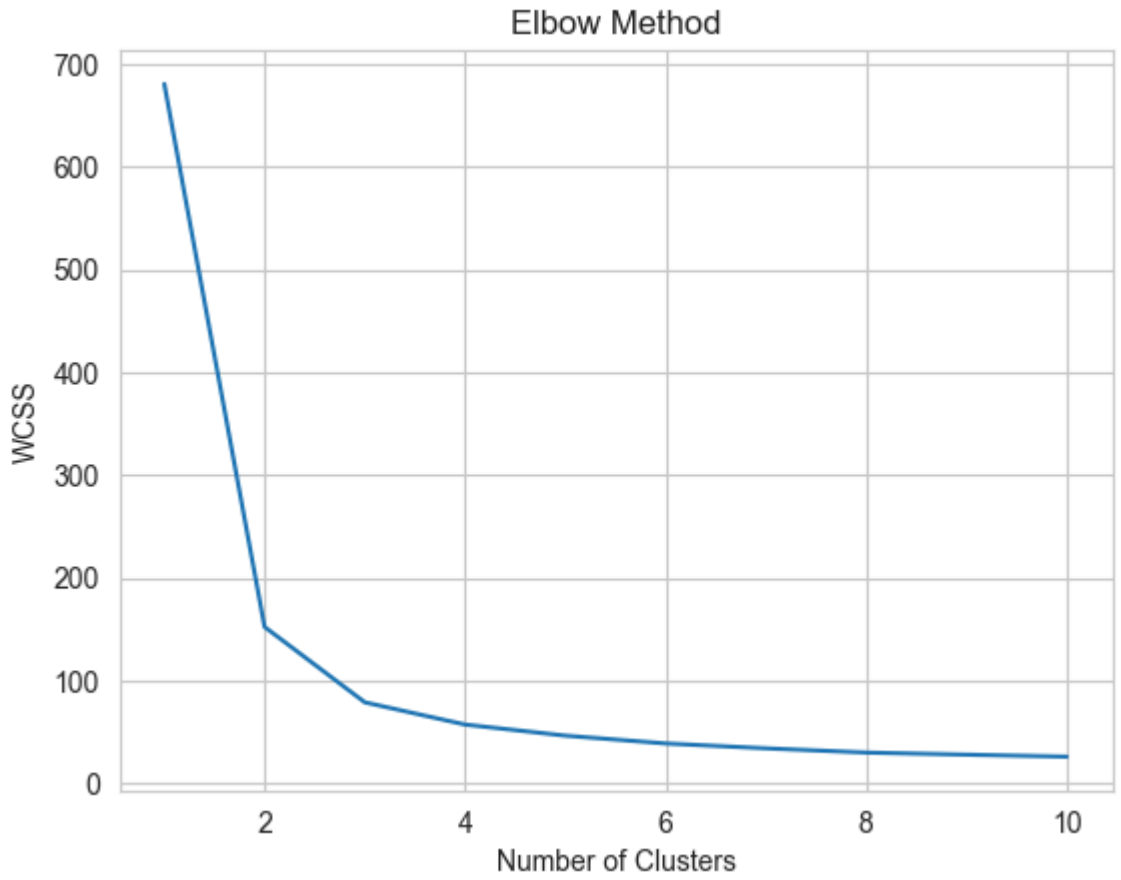

sepal_length
sepal_width
petal_length
petal_width
8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5
2.0
1.5
species
Iris-setosa
Iris-versicolor
Iris-virginica



In [ ]:
```
data = iris.drop(columns=['species'])
```

In [ ]:
```
from sklearn.cluster import KMeans
wcss = []

for i in range(1,11):
 kmeans = KMeans(n_clusters= i , init='k-means++',max_iter=300, n_init= 10 , random_state=0)
 kmeans.fit(data)
 wcss.append(kmeans.inertia_)
```

In [ ]:
```
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```



In [ ]:
```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(data)
```

In [ ]:
```
print("Centroids:")
print(kmeans.cluster_centers_)
```

```
Centroids:
[[5.9016129  2.7483871  4.39354839 1.43387097]
 [5.006      3.418      1.464      0.244     ]
 [6.85       3.07368421 5.74210526 2.07105263]]
```

PCA USING HOUUSING DATSET

In [ ]:
```
import pandas as pd
import numpy as np
```

In [ ]:
```
data = pd.read_csv("newhousing.csv")
```

In [ ]:
```
data.head()
```

Out[ ]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea
0	5250000	5500	3	2	1	1	0	1	0	0	0	0
1	4480000	4040	3	1	2	1	0	0	0	0	1	0
2	3570000	3640	2	1	1	1	0	0	0	0	0	0
3	2870000	3040	2	1	1	0	0	0	0	0	0	0
4	3570000	4500	2	1	1	0	0	0	0	0	0	0

In [ ]:
```
# What type of values are stored in the columns?
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   price             545 non-null    int64
 1   area              545 non-null    int64
 2   bedrooms          545 non-null    int64
 3   bathrooms         545 non-null    int64
 4   stories           545 non-null    int64
 5   mainroad          545 non-null    int64
 6   guestroom         545 non-null    int64
 7   basement          545 non-null    int64
 8   hotwaterheating   545 non-null    int64
 9   airconditioning   545 non-null    int64
 10  parking           545 non-null    int64
 11  prefarea          545 non-null    int64
 12  semi-furnished    545 non-null    int64
 13  unfurnished       545 non-null    int64
 14  areaperbedroom    545 non-null    float64
 15  bbratio           545 non-null    float64
dtypes: float64(2), int64(14)
memory usage: 68.2 KB
```

In [ ]:
```
X = data[['area', 'bedrooms', 'bathrooms', 'stories', 'mainroad',
       'guestroom', 'basement', 'hotwaterheating', 'airconditioning',
       'parking', 'prefarea', 'semi-furnished', 'unfurnished',
       'areaperbedroom', 'bbratio']]

# Putting response variable to y
y = data['price']
```

In [ ]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X=scaler.fit_transform(X)
```

In [ ]:
```
print(X)
```

```
[[0.26460481 0.4        0.33333333 ... 0.         0.23353165 0.6       ]
 [0.16426117 0.4        0.         ... 0.         0.15527684 0.2       ]
 [0.13676976 0.2        0.         ... 1.         0.23138768 0.4       ]
 ...
 [0.45360825 0.4        0.         ... 0.         0.38092941 0.2       ]
 [0.67079038 0.2        0.         ... 0.         0.85608619 0.4       ]
 [0.15931271 0.4        0.         ... 0.         0.1514177  0.2       ]]
```

In [ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state=20)
```

In [ ]:
```
from sklearn.decomposition import PCA
pca = PCA()
X_new = pca.fit_transform(X)
```

In [ ]:
```
pca.get_covariance()
```

Out[ ]:
```
array([[ 0.0222459 ,  0.0033434 ,  0.00484185,  0.00362267,  0.01502118,
         0.00801123,  0.00337738, -0.00028825,  0.01543005,  0.01512004,
         0.01485796,  0.00045307, -0.00996113,  0.01621209,  0.00163042],
       [ 0.0033434 ,  0.02178953,  0.00924492,  0.01743929, -0.00061927,
         0.00455208,  0.00685982,  0.00142337,  0.01102806,  0.00590416,
         0.00494941,  0.00364477, -0.00874798, -0.007912  , -0.00896433],
       [ 0.00484185,  0.00924492,  0.02805286,  0.01579683,  0.00247572,
         0.0081096 ,  0.00816694,  0.00235541,  0.0145631 ,  0.00853796,
         0.0045107 ,  0.0024656 , -0.01038631, -0.00118307,  0.02330959],
       [ 0.00362267,  0.01743929,  0.01579683,  0.08361591,  0.01226952,
         0.00481989, -0.02380599,  0.00114117,  0.03949339,  0.00378253,
         0.00545062, -0.00052055, -0.01126214, -0.00698145,  0.0018242 ],
       [ 0.01502118, -0.00061927,  0.00247572,  0.01226952,  0.12154614,
         0.01232461,  0.00732596, -0.00086009,  0.01709727,  0.02046906,
         0.02956692,  0.00196978, -0.02178562,  0.01175176,  0.00202894],
       [ 0.00801123,  0.00455208,  0.0081096 ,  0.00481989,  0.01232461,
         0.14657312,  0.06802482, -0.00082636,  0.02460874,  0.00411945,
         0.02613667,  0.00109957, -0.01779547,  0.0028055 ,  0.00555808],
       [ 0.00337738,  0.00685982,  0.00816694, -0.02380599,  0.00732596,
         0.06802482,  0.22805586,  0.00043848,  0.01051673,  0.00706287,
         0.0462156 ,  0.01184903, -0.02643686, -0.00109756,  0.00432737],
       [-0.00028825,  0.00142337,  0.00235541,  0.00114117, -0.00086009,
        -0.00082636,  0.00043848,  0.04384781, -0.01266527,  0.00408122,
        -0.0052786 ,  0.00659404, -0.00581827, -0.00061124,  0.00086245],
       [ 0.01543005,  0.01102806,  0.0145631 ,  0.03949339,  0.01709727,
         0.02460874,  0.01051673, -0.01266527,  0.21639234,  0.02126507,
         0.02316851, -0.01220656, -0.02054439,  0.0060774 ,  0.00575573],
       [ 0.01512004,  0.00590416,  0.00853796,  0.00378253,  0.02046906,
         0.00411945,  0.00706287,  0.00408122,  0.02126507,  0.08248111,
         0.01116545,  0.00585649, -0.02233877,  0.0086949 ,  0.00352372],
       [ 0.01485796,  0.00494941,  0.0045107 ,  0.00545062,  0.02956692,
         0.02613667,  0.0462156 , -0.0052786 ,  0.02316851,  0.01116545,
         0.18003238, -0.002415  , -0.01618659,  0.00810504, -0.00150924],
       [ 0.00045307,  0.00364477,  0.0024656 , -0.00052055,  0.00196978,
         0.00109957,  0.01184903,  0.00659404, -0.01220656,  0.00585649,
        -0.002415  ,  0.24347679, -0.13628575, -0.00192466, -0.00122922],
       [-0.00996113, -0.00874798, -0.01038631, -0.01126214, -0.02178562,
        -0.01779547, -0.02643686, -0.00581827, -0.02054439, -0.02233877,
        -0.01618659, -0.13628575,  0.22033864, -0.00394716, -0.00371816],
       [ 0.01621209, -0.007912  , -0.00118307, -0.00698145,  0.01175176,
```

```
         0.0028055 , -0.00109756, -0.00061124,  0.0060774 ,  0.0086949 ,
         0.00810504, -0.00192466, -0.00394716,  0.01820446,  0.00581644],
       [ 0.00163042, -0.00896433,  0.02330959,  0.0018242 ,  0.00202894,
         0.00555808,  0.00432737,  0.00086245,  0.00575573,  0.00352372,
        -0.00150924, -0.00122922, -0.00371816,  0.00581644,  0.03663016]])
```

In [ ]:
```
explained_variance=pca.explained_variance_ratio_
explained_variance
```

Out[ ]:
```
array([0.2295475 , 0.17794124, 0.13925925, 0.09592225, 0.07018757,
       0.06282636, 0.05155462, 0.04543956, 0.04320347, 0.03114886,
       0.02425278, 0.01574126, 0.01217287, 0.00052413, 0.00027827])
```

In [ ]:
```
pca=PCA(n_components=3)
X_new=pca.fit_transform(X)
X_train_new, X_test_new, y_train, y_test = train_test_split(X_new, y, test_size=0.2, random_state=42)
```

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier

knn_pca = KNeighborsClassifier(7)

knn_pca.fit(X_train_new,y_train)
print("Train score after PCA",knn_pca.score(X_train_new,y_train),"%")
print("Test score after PCA",knn_pca.score(X_test_new,y_test),"%")
```

```
Train score after PCA 0.15825688073394495 %
Test score after PCA 0.009174311926605505 %
```

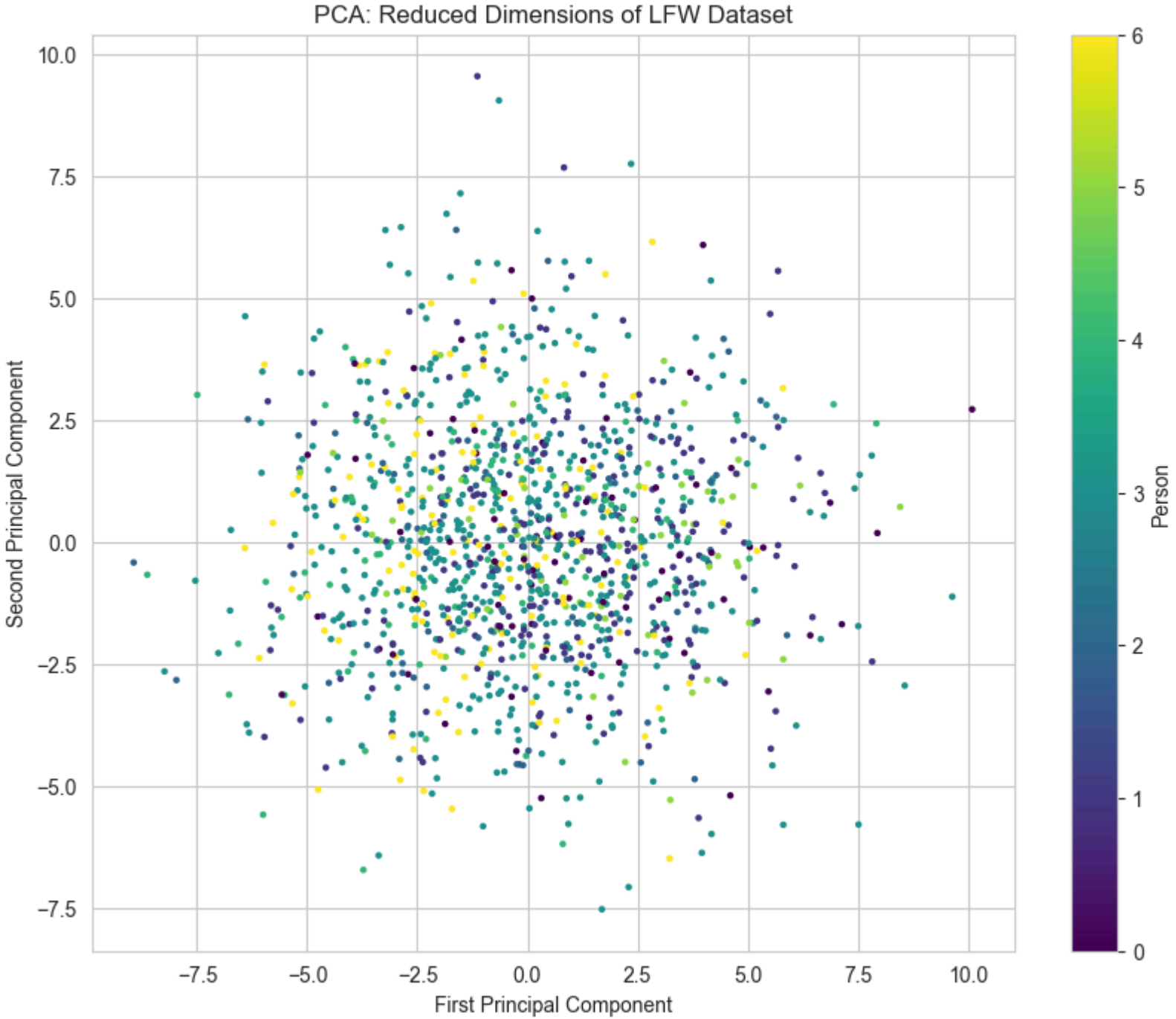
PCA with images

In [ ]:
```
from sklearn.datasets import fetch_lfw_people

lfw_people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
```

In [ ]:
```
# Extract images and labels
X = lfw_people.data
y = lfw_people.target
```

In [ ]:
```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
```

In [ ]:
```
plt.figure(figsize=(10, 8))
scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap='viridis', s=5)
plt.xlabel('First Principal Component')
plt.ylabel('Second Principal Component')
plt.title('PCA: Reduced Dimensions of LFW Dataset')
plt.colorbar(scatter, label='Person')
plt.show()
```


PCA: Reduced Dimensions of LFW Dataset
Second Principal Component
First Principal Component
Person
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
6
5
4
3
2
1
0

In [ ]:
```
import os
import cv2
image_folder=r'C:\Users\Deepthi\OneDrive\Desktop\programs\mlt lab works\mlt lab 8\Cat_Pic'
def load_images(folder, target_size=(300, 300)):
    image_files = sorted([os.path.join(folder, file) for file in os.listdir(folder) if file.endswith(('.jpg', '.png'))])
    images = [cv2.resize(cv2.imread(file), target_size) for file in image_files]
    return np.array(images)

# Load and resize images from the folder
images = load_images(image_folder)
print("Image shape:", images.shape)
```

```
Image shape: (25, 300, 300, 3)
```

In [ ]:
```
plt.figure(figsize=[12,8])
plt.imshow(cv2.cvtColor(images[1], cv2.COLOR_BGR2RGB))  # OpenCV reads images in BGR format, so we convert it to RGB for displ
plt.title('Original Image')
plt.show()
```


Original Image
0
50
100
150
200
250
0
50
100
150
200
250

In [ ]:
```
images_flattened = images.reshape(images.shape[0], -1)
print("Flattened image shape:", images_flattened.shape)
```

```
Flattened image shape: (25, 270000)
```

In [ ]:
```
pca = PCA()
pca.fit(images_flattened)
```

Out[ ]:
```
▾ PCA
PCA()
```

In [ ]:
```
var_cumu = np.cumsum(pca.explained_variance_ratio_) * 100
var_cumu
```

Out[ ]:
```
array([ 22.34683724,  37.60041994,  46.79205472,  54.60804188,
        60.32295644,  65.3722507 ,  69.45593445,  73.24308353,
        76.10939619,  78.73105463,  81.27402831,  83.63704056,
        85.85876853,  87.73831425,  89.46827525,  91.1226531 ,
        92.4911216 ,  93.78163845,  95.00327912,  96.17728962,
        97.27897343,  98.30540158,  99.22200835, 100.        ,
       100.        ])
```

In [ ]:
```
k = np.argmax(var_cumu > 95)
print("Number of components explaining 95% variance:", k)
```

```
Number of components explaining 95% variance: 18
```

In [ ]:
```
from sklearn.decomposition import IncrementalPCA


ipca = IncrementalPCA(n_components=k)
images_recon = ipca.inverse_transform(ipca.fit_transform(images_flattened))
```

In [ ]:
```
# Reshape the reconstructed images
images_recon = images_recon.reshape(images.shape)
images_recon
```

Out[ ]:
```
array([[[[ 3.52100102e+01,  3.14943246e+01,  3.01547374e+01],
         [ 2.94531816e+01,  2.71103233e+01,  2.48999544e+01],
         [ 2.27601252e+01,  1.93329153e+01,  1.71886221e+01],
         ...,
         [ 1.63859138e+02,  1.63400109e+02,  1.19807893e+02],
         [ 1.61793819e+02,  1.61018137e+02,  1.17012614e+02],
         [ 1.61677404e+02,  1.60623053e+02,  1.16655791e+02]],

        [[ 4.36869624e+01,  4.02895239e+01,  3.86166746e+01],
         [ 4.22620659e+01,  3.96633003e+01,  3.78274917e+01],
         [ 3.65152012e+01,  3.34862946e+01,  3.12190489e+01],
         ...,
         [ 1.63777126e+02,  1.63059107e+02,  1.20119104e+02],
         [ 1.63385490e+02,  1.62306579e+02,  1.19281019e+02],
         [ 1.58322929e+02,  1.58281898e+02,  1.15144081e+02]],

        [[ 4.11710477e+01,  4.03606524e+01,  3.63787868e+01],
         [ 4.68378577e+01,  4.62356282e+01,  4.22045559e+01],
         [ 4.48697411e+01,  4.29070674e+01,  3.94815737e+01],
         ...,
         [ 1.64929948e+02,  1.64443478e+02,  1.22929552e+02],
         [ 1.63145859e+02,  1.62469388e+02,  1.19129199e+02],
         [ 1.58554209e+02,  1.57719639e+02,  1.15316283e+02]],

        ...,

        [[ 1.00415654e+02,  1.50035152e+02,  1.66814985e+02],
         [ 1.00901428e+02,  1.50611112e+02,  1.66631373e+02],
         [ 9.43360428e+01,  1.43936507e+02,  1.59958548e+02],
         ...,
         [ 1.06861393e+01,  1.69925965e+02,  1.74683784e+02],
         [ 9.28458307e+00,  1.59627259e+02,  1.64897263e+02],
         [ 1.14845910e+01,  1.51828889e+02,  1.58837186e+02]],

        [[ 9.73999440e+01,  1.40516877e+02,  1.57904204e+02],
         [ 8.88443137e+01,  1.33786879e+02,  1.51253844e+02],
         [ 7.46890108e+01,  1.22203294e+02,  1.38853134e+02],
         ...,
         [ 9.37447267e+00,  1.72844550e+02,  1.76813061e+02],
         [ 7.68176707e+00,  1.64551097e+02,  1.69086423e+02],
```

```
  [ 4.15628710e+00,  1.56494257e+02,  1.61869096e+02]],

 [[ 9.44987276e+01,  1.36591628e+02,  1.54130943e+02],
  [ 8.92104440e+01,  1.32217322e+02,  1.49892379e+02],
  [ 7.98433809e+01,  1.25406538e+02,  1.43107242e+02],
  ...,
  [ 1.20192081e+01,  1.78428526e+02,  1.81084421e+02],
  [ 8.09448321e+00,  1.69847130e+02,  1.73287187e+02],
  [ 5.97304493e+00,  1.61466292e+02,  1.65824023e+02]]],


[[[ 8.78215819e+01,  1.16348701e+02,  1.39912922e+02],
  [ 8.84154039e+01,  1.16703316e+02,  1.40210518e+02],
  [ 8.83538274e+01,  1.16267443e+02,  1.40274570e+02],
  ...,
  [ 5.52688864e+01,  7.70274320e+01,  1.26109899e+02],
  [ 4.43371681e+01,  6.45510989e+01,  1.11992824e+02],
  [ 3.92217603e+01,  5.77812369e+01,  1.04122950e+02]],

 [[ 8.54581964e+01,  1.14099131e+02,  1.38264151e+02],
  [ 9.09580095e+01,  1.19243385e+02,  1.43297214e+02],
  [ 8.74994961e+01,  1.15280370e+02,  1.39627250e+02],
  ...,
  [ 5.81896379e+01,  7.97532187e+01,  1.28234440e+02],
  [ 4.27227652e+01,  6.35650184e+01,  1.10117124e+02],
  [ 3.79454654e+01,  5.70812694e+01,  1.02776969e+02]],

 [[ 8.29107316e+01,  1.11211574e+02,  1.35656579e+02],
  [ 8.65030259e+01,  1.14737653e+02,  1.39114961e+02],
  [ 8.86077699e+01,  1.16398762e+02,  1.40835331e+02],
  ...,
  [ 5.87878712e+01,  8.16006808e+01,  1.29709924e+02],
  [ 4.02337823e+01,  6.05446345e+01,  1.06866827e+02],
  [ 3.29396123e+01,  5.20756379e+01,  9.75228151e+01]],

 ...,

 [[ 5.81953722e+01,  6.45556797e+01,  7.06713955e+01],
  [ 5.71787919e+01,  6.34266620e+01,  6.94750895e+01],
  [ 5.35302602e+01,  5.99860021e+01,  6.59116575e+01],
  ...,
```

```
       [ 6.32874545e+01,  9.11591547e+01,  9.84197469e+01],
       [ 5.96264346e+01,  8.69711017e+01,  9.37404109e+01],
       [ 5.84544908e+01,  8.52261496e+01,  9.20576274e+01]],

      [[ 5.43552700e+01,  6.12932521e+01,  6.74238732e+01],
       [ 5.38129920e+01,  6.00201298e+01,  6.62829375e+01],
       [ 5.09614238e+01,  5.74647327e+01,  6.34959170e+01],
       ...,
       [ 5.99227575e+01,  8.82875830e+01,  9.54806841e+01],
       [ 5.84622310e+01,  8.57415381e+01,  9.25325389e+01],
       [ 5.56835405e+01,  8.27600834e+01,  8.94950669e+01]],

      [[ 5.07439393e+01,  5.67192523e+01,  6.30375865e+01],
       [ 4.95267664e+01,  5.55812007e+01,  6.20649220e+01],
       [ 4.79095136e+01,  5.41313609e+01,  6.05654413e+01],
       ...,
       [ 5.76394739e+01,  8.58573609e+01,  9.31773345e+01],
       [ 5.50159901e+01,  8.21774533e+01,  8.90186230e+01],
       [ 5.56366625e+01,  8.24613941e+01,  8.92323002e+01]]],


     [[[ 2.04193385e+01,  1.93610698e+01,  2.11616114e+01],
       [ 1.98492676e+01,  1.89214916e+01,  2.05460841e+01],
       [ 1.56983844e+01,  1.42970839e+01,  1.60760324e+01],
       ...,
       [ 2.69077288e+01,  4.95244782e+01,  6.40915254e+01],
       [ 2.45306691e+01,  4.79671951e+01,  6.32040818e+01],
       [ 2.61461002e+01,  4.99020301e+01,  6.43718201e+01]],

      [[ 1.67697280e+01,  1.52231200e+01,  1.61985770e+01],
       [ 1.25217267e+01,  1.06245930e+01,  1.18361826e+01],
       [ 1.67999873e+01,  1.49811266e+01,  1.61721121e+01],
       ...,
       [ 2.59341043e+01,  4.85591542e+01,  6.36628394e+01],
       [ 2.40206901e+01,  4.72670125e+01,  6.22597405e+01],
       [ 2.53796841e+01,  4.85862604e+01,  6.29609848e+01]],

      [[ 2.14298468e+01,  1.92410103e+01,  1.98621732e+01],
       [ 2.41794738e+01,  2.23465227e+01,  2.32947967e+01],
       [ 2.51460761e+01,  2.27692954e+01,  2.38488429e+01],
       ...,
```

```
  [ 2.80505817e+01,  4.91316693e+01,  6.46389529e+01],
  [ 2.70609926e+01,  4.91998671e+01,  6.45644004e+01],
  [ 2.81416804e+01,  5.02926703e+01,  6.51780071e+01]],

 ...,

 [[ 7.02691567e+01,  8.38833094e+01,  1.10244255e+02],
  [ 7.21251055e+01,  8.53973451e+01,  1.11427641e+02],
  [ 7.50034746e+01,  8.60227539e+01,  1.12975352e+02],
  ...,
  [ 7.43834740e+01,  8.89433541e+01,  1.06422639e+02],
  [ 6.76451996e+01,  8.19577143e+01,  9.79596394e+01],
  [ 6.70685829e+01,  8.13546935e+01,  9.75377266e+01]],

 [[ 7.06012503e+01,  8.37693472e+01,  1.09459176e+02],
  [ 7.17605974e+01,  8.44182691e+01,  1.10802612e+02],
  [ 7.43705731e+01,  8.44300841e+01,  1.10838711e+02],
  ...,
  [ 9.26925942e+01,  1.07630995e+02,  1.23729697e+02],
  [ 8.79956776e+01,  1.02230739e+02,  1.17382573e+02],
  [ 8.36485737e+01,  9.74933710e+01,  1.13649204e+02]],

 [[ 6.46856413e+01,  7.79383971e+01,  1.03734472e+02],
  [ 6.82550824e+01,  8.16151815e+01,  1.07160033e+02],
  [ 7.36874359e+01,  8.38825703e+01,  1.10112756e+02],
  ...,
  [ 1.05557155e+02,  1.19267296e+02,  1.34703827e+02],
  [ 9.49215643e+01,  1.07610669e+02,  1.23198994e+02],
  [ 9.20195454e+01,  1.04707384e+02,  1.20503357e+02]]],


...,


[[[ 1.22364947e+02,  1.33871473e+02,  1.41036548e+02],
  [ 1.24028187e+02,  1.35847731e+02,  1.42742173e+02],
  [ 1.18513901e+02,  1.29368840e+02,  1.37692595e+02],
  ...,
  [ 9.67879140e+01,  1.17093362e+02,  1.38778732e+02],
  [ 9.43272522e+01,  1.15448412e+02,  1.38914856e+02],
  [ 9.43418889e+01,  1.15597533e+02,  1.39676513e+02]],
```

```
[[ 1.14262099e+02,  1.25361654e+02,  1.31607328e+02],
 [ 1.11171386e+02,  1.21586938e+02,  1.28621537e+02],
 [ 1.18635977e+02,  1.28862528e+02,  1.36466992e+02],
 ...,
 [ 9.51531261e+01,  1.15091103e+02,  1.36677833e+02],
 [ 9.48081531e+01,  1.14730639e+02,  1.37670140e+02],
 [ 9.37406668e+01,  1.14308637e+02,  1.37921061e+02]],

[[ 1.19384032e+02,  1.29491989e+02,  1.35670462e+02],
 [ 1.27484124e+02,  1.37915648e+02,  1.44186364e+02],
 [ 1.30613039e+02,  1.40188920e+02,  1.47420034e+02],
 ...,
 [ 9.64022438e+01,  1.15002268e+02,  1.37202698e+02],
 [ 9.72647434e+01,  1.15740924e+02,  1.38545372e+02],
 [ 9.65496056e+01,  1.15477297e+02,  1.38985444e+02]],

...,

[[ 1.37556103e+02,  1.51616617e+02,  1.64058487e+02],
 [ 1.31979932e+02,  1.46778913e+02,  1.58761238e+02],
 [ 1.31369954e+02,  1.46426280e+02,  1.58402587e+02],
 ...,
 [ 3.42485687e+01,  6.50682583e+01,  8.48126358e+01],
 [ 3.35134243e+01,  6.23974151e+01,  8.27404235e+01],
 [ 3.60862778e+01,  6.37544678e+01,  8.45534576e+01]],

[[ 1.33801986e+02,  1.46986252e+02,  1.58215451e+02],
 [ 1.26544012e+02,  1.40088448e+02,  1.50978650e+02],
 [ 1.25902426e+02,  1.40169073e+02,  1.50821966e+02],
 ...,
 [ 2.20679371e+01,  5.27663106e+01,  7.27668917e+01],
 [ 2.22037309e+01,  5.23477487e+01,  7.32391918e+01],
 [ 2.69844121e+01,  5.64208331e+01,  7.70482074e+01]],

[[ 1.31786331e+02,  1.44053753e+02,  1.54880043e+02],
 [ 1.26708540e+02,  1.38230152e+02,  1.49628111e+02],
 [ 1.26755358e+02,  1.38974534e+02,  1.49508161e+02],
 ...,
 [ 2.17773882e+01,  5.30104957e+01,  7.37503087e+01],
 [ 2.28306984e+01,  5.31125425e+01,  7.49213729e+01],
```

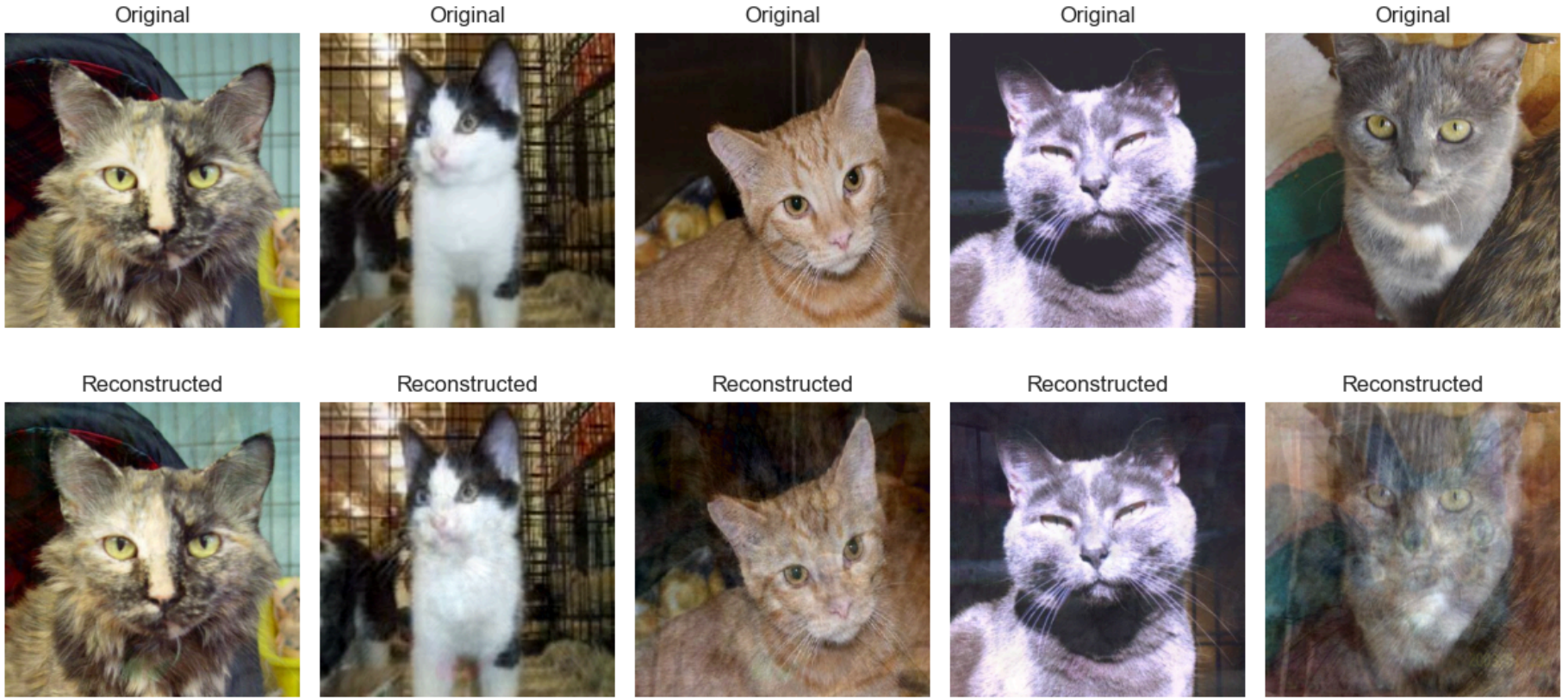
```
  [ 2.95816203e+01,  5.97916068e+01,  8.11078560e+01]]],


[[[ 1.83532211e+02,  1.26769359e+02,  6.69169738e+00],
  [ 1.83177423e+02,  1.26172926e+02,  5.99211177e+00],
  [ 1.85432257e+02,  1.28423610e+02,  7.51805422e+00],
  ...,
  [ 8.01319718e+01,  8.96984336e+01,  9.13192043e+01],
  [ 8.38039861e+01,  9.16908533e+01,  9.43661740e+01],
  [ 8.80591039e+01,  9.32208914e+01,  9.79851523e+01]],

 [[ 1.81597583e+02,  1.24964202e+02,  2.98618376e+00],
  [ 1.84076444e+02,  1.27491607e+02,  5.26280121e+00],
  [ 1.81423741e+02,  1.24802301e+02,  2.06564372e+00],
  ...,
  [ 7.55755591e+01,  8.48688561e+01,  8.81486734e+01],
  [ 7.76263717e+01,  8.50884245e+01,  9.04284298e+01],
  [ 8.47434086e+01,  8.95442136e+01,  9.51797157e+01]],

 [[ 1.78822893e+02,  1.23178913e+02,  4.48239279e-01],
  [ 1.78416607e+02,  1.22681908e+02, -2.35121379e-01],
  [ 1.79095321e+02,  1.23291919e+02, -4.98043128e-02],
  ...,
  [ 6.84862872e+01,  7.75904253e+01,  8.54856335e+01],
  [ 6.92521010e+01,  7.68370025e+01,  8.60876094e+01],
  [ 7.69532616e+01,  8.31024506e+01,  8.96889957e+01]],

 ...,

 [[ 7.77269824e+01,  9.98134959e+01,  1.38986003e+02],
  [ 7.71021250e+01,  9.91681045e+01,  1.37493406e+02],
  [ 6.98094143e+01,  9.19537826e+01,  1.30165626e+02],
  ...,
  [ 2.11546061e+01,  4.42296984e+01,  7.29547784e+01],
  [ 2.24919287e+01,  4.65549611e+01,  7.46337858e+01],
  [ 2.46202434e+01,  4.74338094e+01,  7.51790446e+01]],

 [[ 7.59247653e+01,  9.81464103e+01,  1.36832570e+02],
  [ 7.93213196e+01,  1.01675738e+02,  1.40467307e+02],
  [ 7.60591812e+01,  9.81197040e+01,  1.36993336e+02],
  ...,
```

```
  [ 1.85481378e+01,  4.30986315e+01,  7.12375152e+01],
  [ 2.20218496e+01,  4.55897903e+01,  7.38713808e+01],
  [ 2.41931085e+01,  4.92076664e+01,  7.64111700e+01]],

 [[ 6.80938891e+01,  9.03345185e+01,  1.29275476e+02],
  [ 7.75609621e+01,  9.97834142e+01,  1.38949626e+02],
  [ 7.92288907e+01,  1.01084963e+02,  1.40260649e+02],
  ...,
  [ 1.99003514e+01,  4.62781216e+01,  7.44067065e+01],
  [ 2.13836899e+01,  4.57522673e+01,  7.43086868e+01],
  [ 2.70790144e+01,  5.20378720e+01,  8.05125880e+01]]],


[[[ 1.09945290e+02,  1.41847899e+02,  2.06508441e+02],
  [ 1.10617576e+02,  1.43290704e+02,  2.08147082e+02],
  [ 1.09206127e+02,  1.42774491e+02,  2.11246894e+02],
  ...,
  [ 6.58647972e+01,  7.47994296e+01,  8.32717489e+01],
  [ 6.67362171e+01,  7.53093482e+01,  8.27273838e+01],
  [ 7.03839022e+01,  7.78690069e+01,  8.50694988e+01]],

 [[ 1.11371393e+02,  1.43730309e+02,  2.07814309e+02],
  [ 1.03886910e+02,  1.37017107e+02,  2.01193389e+02],
  [ 1.11891556e+02,  1.45627181e+02,  2.12762637e+02],
  ...,
  [ 6.38254633e+01,  7.24337136e+01,  8.17126335e+01],
  [ 6.74428332e+01,  7.55160620e+01,  8.39235481e+01],
  [ 6.62809592e+01,  7.35150542e+01,  8.16415396e+01]],

 [[ 1.08196350e+02,  1.40724406e+02,  2.04259247e+02],
  [ 1.07680950e+02,  1.40637707e+02,  2.04197037e+02],
  [ 1.08914188e+02,  1.42363527e+02,  2.08910364e+02],
  ...,
  [ 6.50816449e+01,  7.31584929e+01,  8.34052619e+01],
  [ 6.63794597e+01,  7.41356206e+01,  8.35192928e+01],
  [ 6.56171200e+01,  7.26750608e+01,  8.19285308e+01]],

 ...,

 [[ 1.89668256e+02,  1.92336082e+02,  1.88318731e+02],
  [ 1.83777636e+02,  1.86513353e+02,  1.82529116e+02],
```

```
        [ 1.76937754e+02,  1.79603575e+02,  1.75709040e+02],
        ...,
        [ 1.15308761e+02,  1.14152901e+02,  1.41383383e+02],
        [ 1.18586402e+02,  1.18428138e+02,  1.42493819e+02],
        [ 1.21329960e+02,  1.21454671e+02,  1.45564049e+02]],

       [[ 1.90172999e+02,  1.93090716e+02,  1.89365167e+02],
        [ 1.87192511e+02,  1.90008460e+02,  1.86269373e+02],
        [ 1.81563960e+02,  1.84218075e+02,  1.80614684e+02],
        ...,
        [ 1.13558487e+02,  1.13283798e+02,  1.37815054e+02],
        [ 1.15348483e+02,  1.14950425e+02,  1.39322000e+02],
        [ 1.18311000e+02,  1.16915042e+02,  1.41329950e+02]],

       [[ 1.11540967e+02,  1.14522347e+02,  1.10851700e+02],
        [ 1.08902684e+02,  1.11797124e+02,  1.08137305e+02],
        [ 1.04901170e+02,  1.07462781e+02,  1.03958011e+02],
        ...,
        [ 5.71598560e+01,  5.69496450e+01,  8.15004754e+01],
        [ 5.94006282e+01,  5.87595404e+01,  8.34306409e+01],
        [ 6.21571109e+01,  6.06090554e+01,  8.42852770e+01]]]])
```

In [ ]:

```python
# Plot some of the original and reconstructed images
n_images = 5  # Number of images to plot
plt.figure(figsize=[12, 6])
for i in range(n_images):
    plt.subplot(2, n_images, i + 1)
    plt.imshow(cv2.cvtColor(images[i], cv2.COLOR_BGR2RGB))
    plt.title('Original')
    plt.axis('off')

    plt.subplot(2, n_images, n_images + i + 1)
    reconstructed_image = cv2.convertScaleAbs(images_recon[i])  # Convert reconstructed image to compatible format
    plt.imshow(cv2.cvtColor(reconstructed_image, cv2.COLOR_BGR2RGB))
    plt.title('Reconstructed')
    plt.axis('off')

plt.tight_layout()
plt.show()
```


Original
Original
Original
Original
Original
Reconstructed
Reconstructed
Reconstructed
Reconstructed
Reconstructed

In [ ]: