Практические задания №2 (6304 Григорьев И.С.)

Задание 1

In [54]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import KernelPCA
```

In [39]:
```
X = np.array([
    [4, 2.9],
    [2.5, 1],
    [3.5, 4],
    [2, 2.1]
])
```

In [40]:
```
def kernel_function(Xi, Xj):
    return round(sum((Xi - Xj) ** 2), 2)
```

In [56]:
```
kernel_matrix = np.array([[kernel_function(Xi, Xj) for Xj in X] for Xi in X])
kernel_matrix
```

Out[56]:
```
array([[ 0.  ,  5.86,  1.46,  4.64],
       [ 5.86,  0.  , 10.  ,  1.46],
       [ 1.46, 10.  ,  0.  ,  5.86],
       [ 4.64,  1.46,  5.86,  0.  ]])
```

Задание 2

In [42]:
```
data = np.array([
    [8, -20],
    [0, -1],
    [10, -19],
    [10, -20],
    [2, 0]
])
```

In [43]:
```
data_mean = np.mean(data, axis=0)
print('Среднее значение:', data_mean)
```

```
Среднее значение: [  6. -12.]
```

In [44]:
```
cov_matrix = np.cov(data.T)
print('Ковариационная матрица:\n', cov_matrix)
```

```
Ковариационная матрица:
 [[ 22.  -47.5]
 [-47.5 110.5]]
```

In [45]:
```
eigvals = np.linalg.eigvals(cov_matrix)
print('Собственные числа:', eigvals)
```

```
Собственные числа: [  1.33226359 131.16773641]
```

In [46]:
```
print('"Внутренний" размер набора данных:', data.shape)
```

```
"Внутренний" размер набора данных: (5, 2)
```

In [47]:
```
centered_data = data - data_mean
centered_data
```

Out[47]:
```
array([[ 2., -8.],
       [-6., 11.],
       [ 4., -7.],
       [ 4., -8.],
       [-4., 12.]])
```

In [48]:
```
eigvals, eigvecs = np.linalg.eig(cov_matrix)
eigvals, eigvecs
```

Out[48]:
```
(array([  1.33226359, 131.16773641]),
 array([[-0.91696017,  0.39897876],
        [-0.39897876, -0.91696017]]))
```

In [49]:
```
eig_vector = - eigvecs[:, np.argmax(eigvals)]
eig_vector
```

Out[49]:
```
array([-0.39897876,  0.91696017])
```

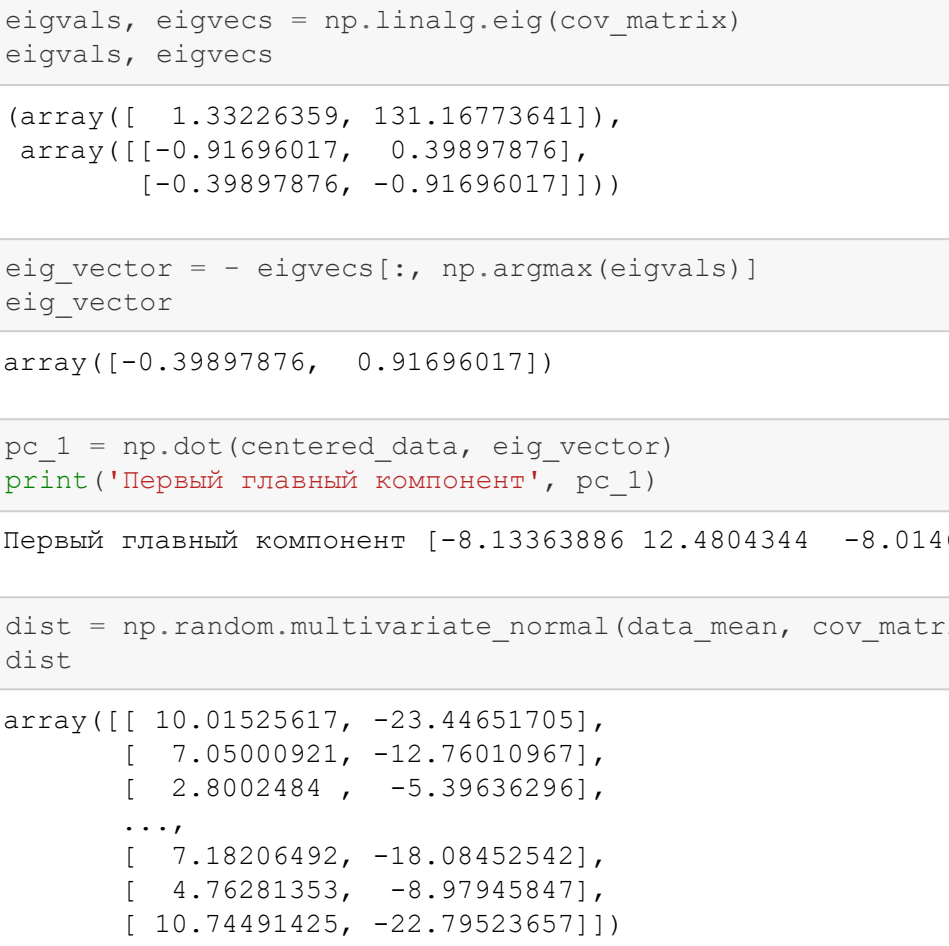
In [50]:
```
pc_1 = np.dot(centered_data, eig_vector)
print('Первый главный компонент', pc_1)
```

```
Первый главный компонент [-8.13363886 12.4804344  -8.01463621 -8.93159638 12.59943705]
```

In [51]:
```
dist = np.random.multivariate_normal(data_mean, cov_matrix, size=5000)
dist
```

Out[51]:
```
array([[ 10.01525617, -23.44651705],
       [  7.05000921, -12.76010967],
       [  2.8002484 ,  -5.39636296],
       ...,
       [  7.18206492, -18.08452542],
       [  4.76281353,  -8.97945847],
       [ 10.74491425, -22.79523657]])
```

In [52]:
```
x, y = dist.T
sns.displot(x=x, y=y, kind='kde')
plt.show()
```



Задание 3

In [62]:
```
transform = kernel_matrix / 100 + np.ones((X.shape[0], X.shape[0])) * 0.5
transformed_kernel_matrix = transform @ kernel_matrix @ transform
precomputed_data = KernelPCA(1, kernel='precomputed').fit_transform(transformed_kernel_matrix)
precomputed_data
```

Out[62]:
```
array([[-0.05127142],
       [ 0.05127142],
       [ 0.05127142],
       [-0.05127142]])
```