Лабораторная работа №7: «Метод главных компонент»

Набор данных **ex7data1.mat** представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит две переменные X_1 и X_2 - координаты точек, для которых необходимо выделить главные компоненты.

Набор данных **ex7faces.mat** представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит 5000 изображений 32x32 в оттенках серого. Каждый пиксель представляет собой значение яркости (вещественное число). Каждое изображение сохранено в виде вектора из 1024 элементов. В результате загрузки набора данных должна быть получена матрица 5000x1024.

Задание.

1. Загрузите данные **ex7data1.mat** из файла.
2. Постройте график загруженного набора данных.
3. Реализуйте функцию вычисления матрицы ковариации данных.
4. Вычислите координаты собственных векторов для набора данных с помощью сингулярного разложения матрицы ковариации (разрешается использовать библиотечные реализации матричных разложений).
5. Постройте на графике из пункта 2 собственные векторы матрицы ковариации.
6. Реализуйте функцию проекции из пространства большей размерности в пространство меньшей размерности с помощью метода главных компонент.
7. Реализуйте функцию вычисления обратного преобразования.
8. Постройте график исходных точек и их проекций на пространство меньшей размерности (с линиями проекций).
9. Загрузите данные **ex7faces.mat** из файла.
10. Визуализируйте 100 случайных изображений из набора данных.
11. С помощью метода главных компонент вычислите собственные векторы.
12. Визуализируйте 36 главных компонент с наибольшей дисперсией.
13. Как изменилось качество выбранных изображений?
14. Визуализируйте 100 главных компонент с наибольшей дисперсией.
15. Как изменилось качество выбранных изображений?
16. Используйте изображение, сжатое в лабораторной работе №6 (Кластеризация).
17. С помощью метода главных компонент визуализируйте данное изображение в 3D и 2D.
18. Соответствует ли 2D изображение какой-либо из проекций в 3D?
19. Ответы на вопросы представьте в виде отчета.

Реализация:

In[1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pnd
```

1.1 Загрузите набор данных ex7data1.mat из файла.

In[2]:

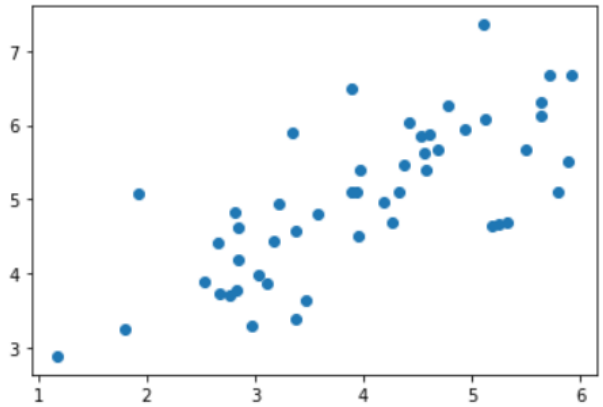
```
mat = loadmat('data/ex7data1.mat')
X = mat['X']
X.shape
```

```
(50, 2)
```

1.2 Постройте график загруженного набора данных.

In[3]:

```
plt.scatter(X[:,0], X[:,1])
```



1.3 Реализуйте функцию вычисления матрицы ковариации данных

In[4]:

```
def cov_matrix(X):
    return np.dot(X.T, X) / X.shape[0]
```

1.4 Вычислите координаты собственных векторов для набора данных с помощью сингулярного разложения матрицы ковариации (разрешается использовать библиотечные реализации матричных разложений)

In[5]:

```
def normalize_features(X):
    mu = np.mean(X, axis=0)
    sigma = np.std(X, axis=0)

    X_norm = (X - mu)/sigma

    return X_norm, mu , sigma

from numpy.linalg import svd

def pca(X):
    sigma = cov_matrix(X)
    return svd(sigma)

X_norm, mu, std = normalize_features(X)
U, S, V = pca(X_norm)
U
```

Out[5]:

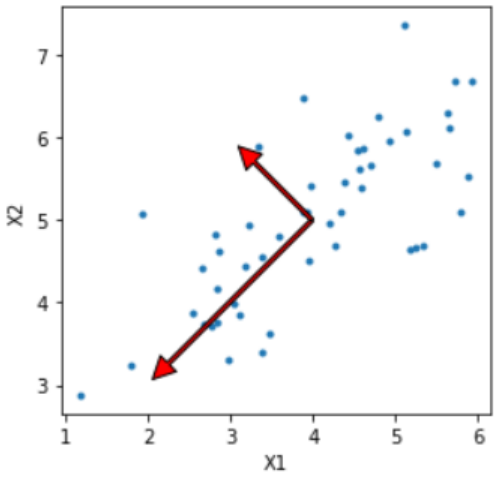
```
array([[-0.70710678, -0.70710678],
       [-0.70710678,  0.70710678]])
```

1.5 Постройте на графике из пункта 2 собственные векторы матрицы ковариации

In[7]:

```
mu = X.mean(axis=0)
projected_data = np.dot(X, U)
sigma = projected_data.std(axis=0).mean()

fig, ax = plt.subplots()
ax.plot(X[:, 0], X[:, 1], marker='o', linestyle="None", markersize=3)
for ind, axis in enumerate(U):
    start, end = mu, mu + (S[ind] + sigma) * axis
    ax.annotate(
        '', xy=end, xycoords='data',
        xytext=start, textcoords='data',
        arrowprops=dict(facecolor='red', width=2.0))
ax.set_aspect('equal')
ax.set_xlabel('X1')
ax.set_ylabel('X2')
plt.show()
```



In[8]:

```
U[:,0]
```

Out[8]:

```
array([-0.70710678, -0.70710678])
```

1.6. Реализуйте функцию проекции из пространства большей размерности в пространство меньшей размерности с помощью метода главных компонент

In[9]:

```
def make_projection(X, U, K):
    U_reduce = U[:, :K]
    return np.dot(X, U_reduce)

Z = make_projection(X_norm, U, 1)
Z.shape
```

Out[10]:

```
(50, 1)
```

1.7 Реализуйте функцию вычисления обратного преобразования

In[11]:

```
def recover(Z, U, K=None):
    U_reduce = U[:, :K]
    return np.dot(Z, U_reduce.T)
In [143]:
X_rec  = recover(Z, U, 1)
```

1.8 Постройте график исходных точек и их проекций на пространство меньшей размерности (с линиями проекций)

In[12]:

```
plt.scatter(X_norm[:,0],X_norm[:,1],marker="o",label="Original",facecol-
ors="none",edgecolors="b",s=15)
plt.scatter(X_rec[:,0],X_rec[:,1],marker="o",label="Approximation",facecol-
ors="none",edgecolors="r",s=15)
plt.title("The Normalized and Projected Data after PCA")
plt.legend()
```