Assignment4

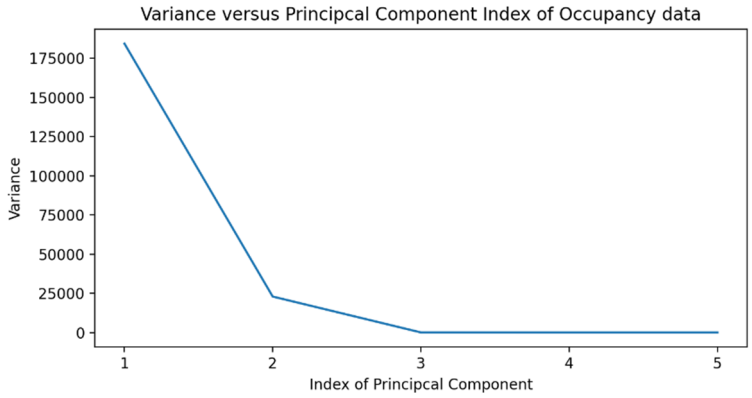
Exercise 1

a)

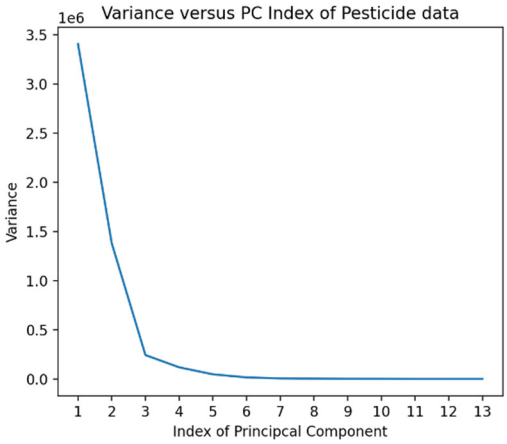
```
#calculate the eigenvalue and eigenvector
def pca(data):
    cov_matrix = np.cov(data.T) # the definition of covariance matrix
    evals, evecs = np.linalg.eigh(cov_matrix)
    evals = evals[::-1]
    evecs = evecs[:,::-1]
    return evals, evecs
```

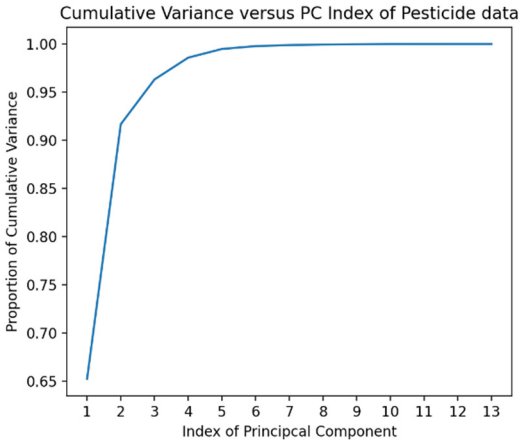
The covariance matrix calculates the covariance among different dimensions. So, the data should be transposed at the beginning. Then, I used “np.linalg.eigh()” to get eigenvalue and eigenvectors. Because “np.linalg.eigh()” returns the eigenvalue (and corresponding eigenvectors) in ascending order, I used the “slice tool” to get the eigenvalue (and corresponding eigenvectors) in descending order.

b)



c)





d)

for pesticide data, the needed number of PCs to capture 90% of the variance is 2; the needed number of PCs to capture 95% of the variance is 3.

Exercise 2

a)

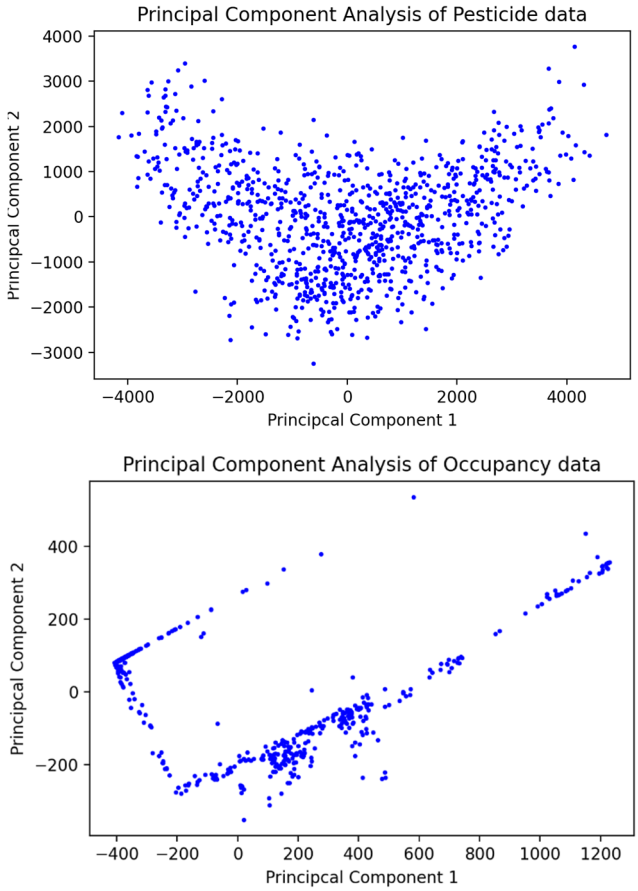
```
def pca_trans(data, n):
    cov_matrix = np.cov(data.T) # the definition of covariance matrix
    evals, evecs = np.linalg.eigh(cov_matrix)
    evals = evals[::-1]
    evecs = evecs[:, ::-1]
    evecs = evecs[:, :n]
    data_tran = evecs.T @ data.T
    data_tran = data_tran.T
    return data_tran
```

or

```
def pca_trans(data, n):
    modelPCA2 = PCA(n_components=n)
    Xtrans = modelPCA2.fit_transform(data)
    return Xtrans
```

Keep the first n eigenvectors (in this exercise, n=2), then the eigenvector matrix should be transposed and do a matrix multiplication with transposed data. Finally, the data should be transposed again to gain the result.

b)



Exercise 3

Yes, applying these preprocessing steps will make a difference. PCA is sensitive to the relative scaling of the original data.

Centering is sort of moving the “original point of coordinate axis” to the center of data. It could

“detrend” the data, only showing the difference.

Standardizing is like equalizing the weight of different features. It eliminates the effect of different feature scales.

Exercise 4

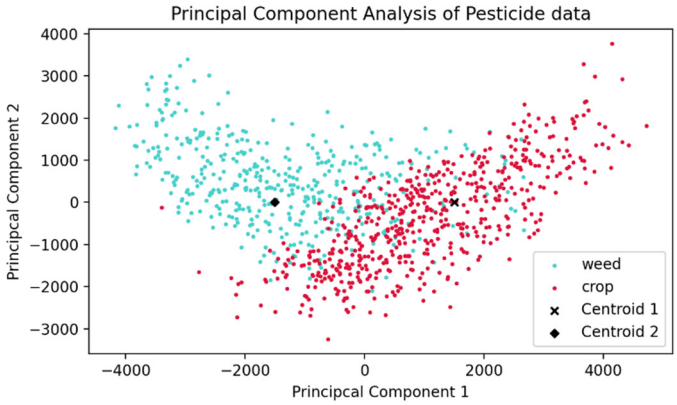
a)

Projection of the two cluster centers:

The first is [1.50515072e+03, 3.01448777e-13]

The second is [-1.50515072e+03, 3.01448777e-13]

b)



c)

After PCA, the dimension of features decreased from 13 to 2. The plot shows that most of the pesticide data on different labels are distributed separately, but some overlaps exist. The locations of the two cluster centers are proper in general. So, the pesticide data after PCA kept the main difference of features, and the clusters are meaningful to some extent.

Exercise 5

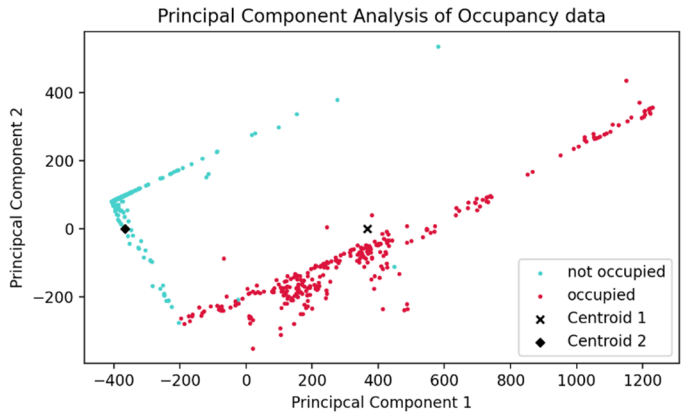
a)

Projection of the two cluster centers:

The first is [3.66874141e+02, 8.64415398e-14]

The second is [-3.66874141e+02, 8.64415398e-14]]

b)



c)

After PCA, the dimension of features decreased from 5 to 2. The plot shows that almost all occupancy data in different labels are distributed separately. The locations of the two cluster centers are proper in general. So, the occupancy data after PCA kept the main difference of features, and the clusters are meaningful.

Exercise 6

a)

```
def multivarlinreg(X, t):
    row = len(X)
    col_1 = np.ones(row)
    X = np.c_[col_1, X] # add a column of ones at the first column
    w = np.linalg.inv(X.T @ X) @ X.T @ t
    return w
```

According to the formula "$w = (X^TX)^{-1}X^T y$", the function is defined. And because the offset w_0 (x_0 should equal 1), I added a column of ones at the first column of the X matrix before calculating w.

b)

w_0= -0.256, w_1=189.245.

(w_0-w_1: intercept, humidity)

When only using the "humidity" feature and labels to perform regression, the intercept of the multivariable linear regression is -0.256. And the occupancy status has a positive correlation of humidity feature with a coefficient of 189.245, which means with the increase of humidity, the occupancy status tends to be 1.

c)

w_0= 4.105, w_1= -0.202, w_2= -9.778e-02, w_3= 2.201e-03, w_4= 2.639e-05, w_5= 6.566e+02.

(w_0-w_5: intercept, time, temperature, light, CO_2, humidity)

When using all five features and labels to perform regression, the intercept of the multivariable linear regression is 4.105. And the occupancy status has a negative correlation with both the time and temperature, which means with the increase of time and temperature, the occupancy status tends to be 0. Regardless of the "sign", the coefficient w_1 is larger than w_2, which indicates that time has a stronger negative impact on occupancy status.

In addition, the occupancy status has a positive correlation with light, CO_2, and humidity features, which means with the increase of light, CO_2, and humidity, the occupancy status tends to be 1. The coefficient $w_5>w_3>w_4$, which means humidity has the strongest positive impact on occupancy status while CO_2 has the weakest.

The results of c) and b) are quite different. It makes sense because I used different independent variables to perform regression.

Exercise 7

a)

```
def rmse(prediciton, true):
    error = (true - prediction)**2
    rmse = np.sqrt(np.mean(error))
    return rmse
```

or

```
def test(weights, test_x, test_y):
    row = len(test_x)
    col_1 = np.ones(row)
    X = np.c_[col_1, test_x] # add a column of ones at the first column
    prediction = X @ weights
    error = (test_y - prediction)**2
    rmse = np.sqrt(np.mean(error))
    return rmse
```

The function was defined as above according to the given formula. RMSE for the humidity is 0.488, which is high.

b)

RMSE for all features is 0.183, which is much less than the result in a). So, with the increasing of the variables from one to five, the regression performance gets better.