Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Штучного інтелекту

Звіт з лабораторної роботи №3

з курсу Інтелектуальний аналіз даних

по темі “КОМПРЕСІЯ (СТИСК) ІНФОРМАЦІЇ МЕТОДОМ

ГОЛОВНИХ КОМПОНЕНТ (РСА) І НА ОСНОВІ СИСТЕМИ Е. ОЯ.”

Виконав:

студент гр. ІТШI 17-1

Петрикiн Михайло

Перевірила:

доц. Дейнеко А.О.

Харків 2019

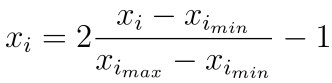
1. Цель работы: Приобретение навыков реализации компрессии данных методом PCA.

2. Ход работы:

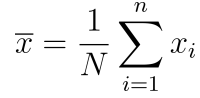
2.1: Предварительная подготовка данных

2.1.1. Массив исходных данных разделить на атрибуты. Каждый атрибут кодировать на гиперкуб [-1; 1]:





2.1.2. Найти среднее значение



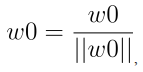
2.1.3. Отцентрировать все значения относительно среднего:



2.1.4. Снова собрать атрибуты в массив X.

2.2. Вычисление:

Необходимо вычислить *c* векторов *w* (матрица *W*) и *с* компонент *y* (матрица *Y*). Векторы и компоненты вычисляются рекуррентно, то есть на основе предыдущих значений. «Стартовый» вектор *w0* весовых коэффициентов состоит из *с* случайных чисел в диапазоне [-1, 1]. Стартовый вектор *w0* нужно нормировать:

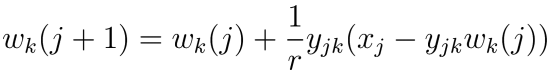


Компонента *у* состоит из *r* скаляров. Каждый следующий скаляр вычисляется из вектора весовых коэффициентов *w* и значений *х* по формуле:

,

где j = 1:r, k = (1,c), x(j) - образец из массива данных (строка длины *с*).

Обратите внимание, что в данной формуле используется матричное умножение. Поэтому необходимо, чтобы w был представлен строкой, а x(j) - столбцом либо вектором(векторное умножение). Тогда результатом умножения будет число. На основе полученного y(j,k) и предыдущего значения w(j) вычислим wk (j+1) по правилу Оя:

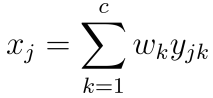


После каждого пересчета вектор нормируем. Обратите внимание: w(k) пересчитывается r раз, но нам необходимо только последнее значение - оно является k-тым вектором, остальные хранить не нужно. При этом компонента у содержит все r значений. Для стабилизации результата компоненту y(k) и вектор w(k) следует вычислить 10^k раз. Далее после того как будет получен каждый следующий собственный вектор и главные компоненты необходимо производить процедуру вычитания главных компонент и собственных векторов из выборки:



2.3. Восстановление данных:

Для проверки правильности сжатия (компресии) необходимо воспользоваться соотношением вида:

,

где x(j) - строка исходного массива данных.

Также в качестве проверки можно сравнить полученный результат с результатом встроенного метода PCA. Сравнивать стоит по матрице W. При этом знаки могут не совпадать.

3. Код основных этапов реализации метода Э. Оя:

def Scal(a,b):

scal=0

for i in range(len(a)):

scal+=a[i]\*b[i]

return scal

def sub(a,b):

x=[0. for i in range(len(a))]

for i in range(len(x)):

x[i]=a[i]-b[i]

return x

def mulNum(a,b):

x=[0. for i in range(len(b))]

for i in range(len(x)):

x[i]= a\*b[i]

return x

def dev(a,b):

x=[0. for i in range(len(a))]

for i in range(len(x)):

x[i]=a[i]/b

return x

def S(a,b):

x=[0. for i in range(len(a))]

for i in range(len(x)):

x[i]=a[i]+b[i]

return x

def rec(a,b):

x=[[0. for i in range(len(a[0]))]for j in range(len(a))]

x=np.array(x)

for k in range(len(a[0])):

z=[[0. for i in range(len(a[0]))]for j in range(len(a))]

z=np.array(z)

for i in range(len(a)):

z[i]=mulNum(a[i,k],b[k])

for i in range(len(a)):

for j in range(len(a[0])):

x[i,j]+=z[i,j]

return x

def Oj(x):

Y=[[0.for i in range(len(x[0]))]for j in range(len(x))]

w=[[0.for i in range(len(x[0]))]for j in range(len(x[0]))]

Y=np.array(Y)

w=np.array(w)

for i in range(len(w)):

w[0,i]=random.uniform(-1,1)

for i in range(len(Y)):

Y[i,0]=Scal(w[0],x[i])

for k in range(len(x[0])-1):

W=[[0. for i in range(len(x[0]))]for j in range(len(x))]

for z in range(10000):

W[0]=Norm(w[k])

for i in range(1,len(W)):

W[i]= S(W[i-1], mulNum(Y[i-1,k], dev(sub(x[i-1], mulNum(Y[i-1,k], W[i-1])), i)))

Norm(W[i])

w[k+1]=W[len(W)-1]

for i in range(len(Y)):

Y[i,k+1]=Scal(w[k+1],x[i])

return Y

Реализация методов для выборки iris:

data=datasets.load\_iris()

x=data.data

HyperCube(x)

averData=Aver(x)

Center(x,averData)

col=data.target

Y=Oj(x)

plt.scatter(Y[:,0],Y[:,1],c=col,s=20,cmap='Set1')

plt.title('Ирисы',fontsize=20)

plt.xlabel('1 главная компонента',fontsize=15)

plt.ylabel('2 главная компонента',fontsize=15)

plt.show()

plt.scatter(Y[:,2],Y[:,3],c=col,s=20,cmap='Set1')

plt.title('Ирисы',fontsize=20)

plt.xlabel('3 главная компонента',fontsize=15)

plt.ylabel('4 главная компонента',fontsize=15)

plt.show()

4. Скриншоты:

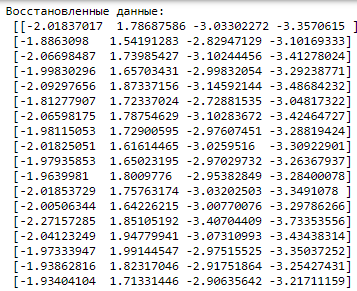


Рис. 1 - Данные выборки Iris после восстановления.

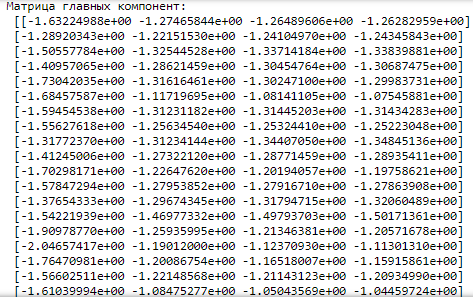


Рис. 2 - Матрица главные компонент Iris.

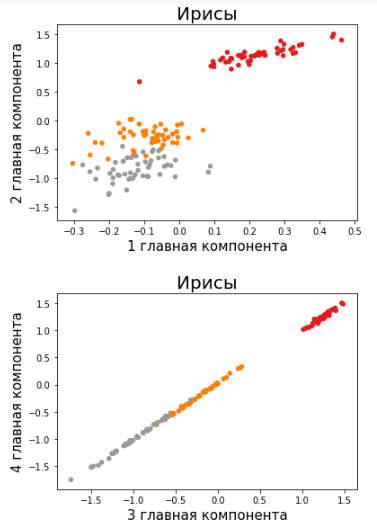


Рис. 3 - Зависимость главных компонент в выборке Iris.

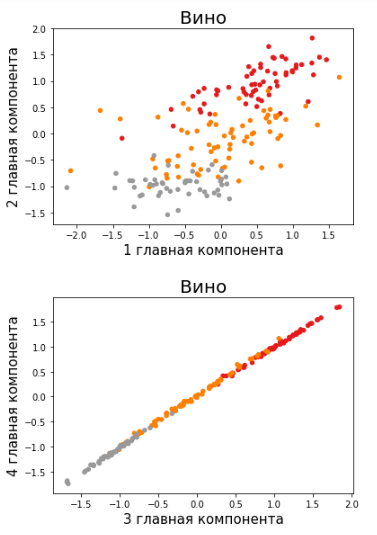


Рис. 4 - Зависимость главных компонент в выборке Wine.

5. Выводы: В ходе данной лабораторной работы была повторена и отточена теория по методу PCA Э. Оя, реализован метод с помощью средств языка программирования python, приобретен новый полезный навык для предобработки данных, а именно - компрессия данных, благодаря которой, в нашем случае, вся полезная информация переносится на первые две главные компоненты и упрощает дальнейшее выполнение основных задач анализа данных.