

**سهیل عبداللهی 9823063**

**مهشید جعفری 9923015**

**تمرین چهارم آز هوش محاسباتی**

**استاد: جناب حیدریان**

**سوال1)**

**کتابخانه های لازم را ایمپورت کرده و بکمک دستور make\_blobs تعداد 200 نقطه در 4 دسته تولید میکنیم و نتیجه را میبینیم:**

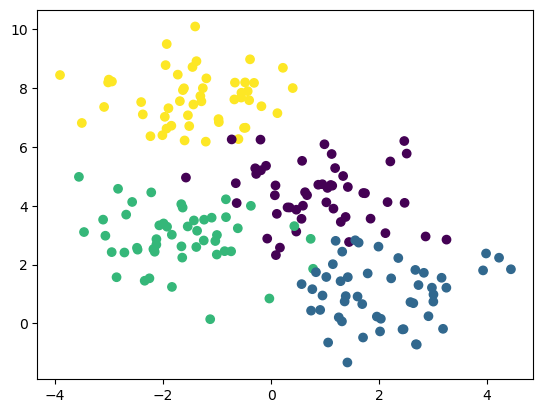
import numpy as np

from sklearn.datasets import make\_blobs

X, y = make\_blobs(n\_samples=200, centers=4, n\_features=2,random\_state=0)

import matplotlib.pyplot as plt

plt.scatter(X[:, 0], X[:, 1], c=y)

****

**حال کلاس kmeans را ایجاد و مدل را طراحی میکنیم. کلاس kmeans را ایجاد کرده و در متد init، متغیر های تعداد کلاس و تعداد دفعات ترین و خطا و لیست ارورها را تعریف میکنیم.**

**متد بعدی، fit است که توضیح آن به شرح زیر است:**

**از بین داده های ورودی، 4 داده بصورت رندوم، بعنوان مرکز خوشه انتخاب میشوند.**

**همچنین داده ها را در متغیر دیگری به اسم picture میریزیم. حال در حلقه تکرار آموزش، متغیر cluster را که لیستی شامل 4 لیست خالی است را میسازیم.**

**سپس در حلقه دیگری، درایه هارا در لیستی میریزیم که نزدیک ترین فاصله به مرکز خوشه آن را دارد. نهایتا میانگین اعضای هر لیست را بعنوان مرکز دسته جدید در نظر میگیریم و این کار تکرار میشود.**

**در هر مرحله، فاصله درایه ها از مرکز دسته آن مرحله را در لیست error میریزیم. سپس میانگین ارور های هر دسته را محاسبه کرده و در به لیست all\_error اضافه میکنیم.**

class kmeans:

    def \_\_init\_\_(self,n\_classes):

        self.number\_of\_classes=n\_classes

        self.max\_iter=100

        self.tolorance=1e-8

        self.all\_error=[[],[],[],[]]

    def fit(self,x):

        self.random\_center\_index=np.random.choice(range(len(x)),4)

        self.random\_center=x[self.random\_center\_index]

        for j in range(self.max\_iter):

            self.clustered=[[],[],[],[]]

            self.error=[[],[],[],[]]

            for i in range(200):

                distance=np.linalg.norm(self.random\_center-x[i],axis=-1)

                minimum\_index=np.argmin(distance)

                self.clustered[minimum\_index].append(x[i])

                self.error[minimum\_index].append(distance)

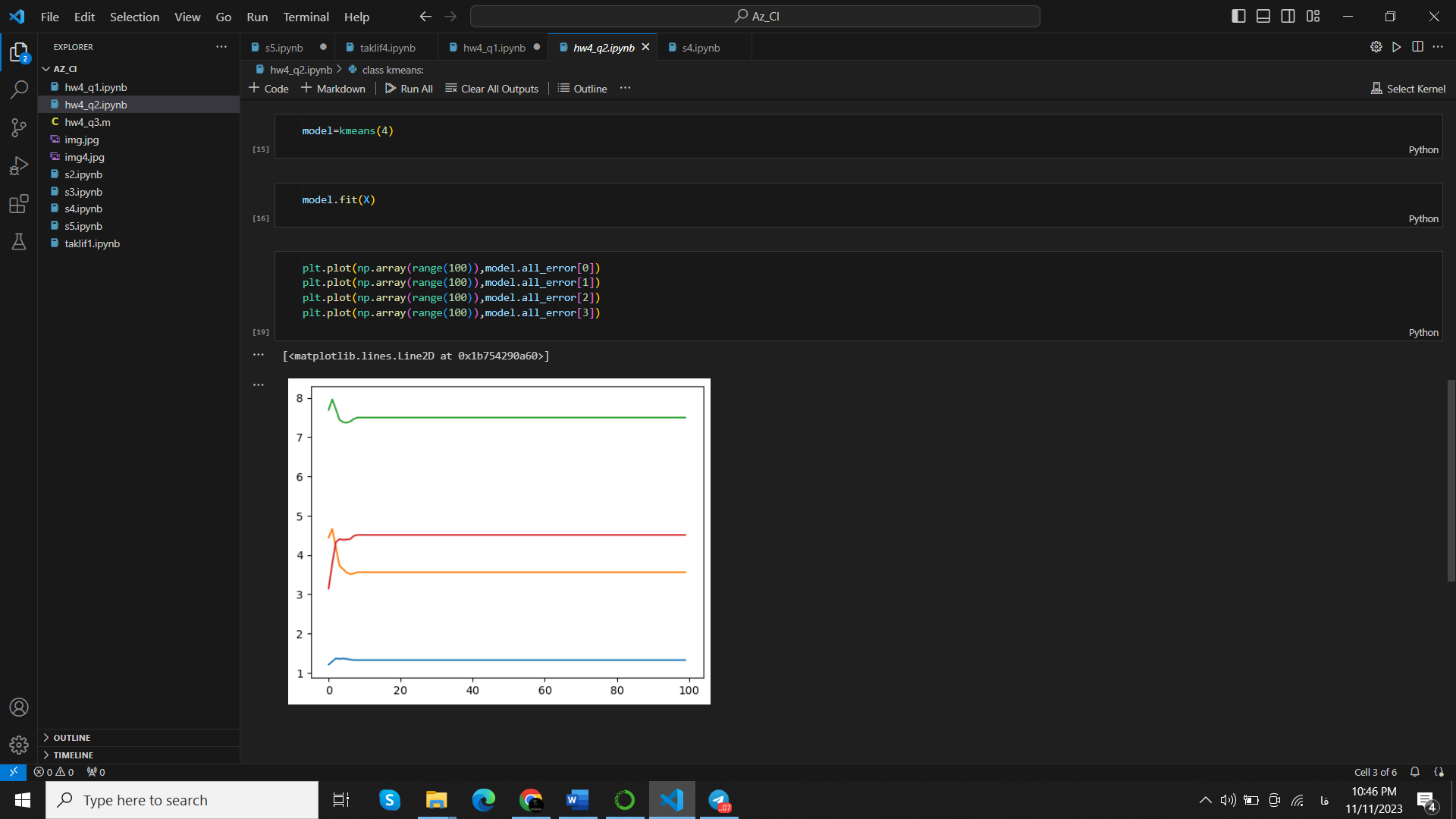
            for i in range(4):

                self.random\_center[i]=np.mean(self.clustered[i], axis=0)

                self.error[i]=np.mean(self.error[i], axis=0)

                self.all\_error[i].append(self.error[i][0])

**حال از مدل استفاده میکنیم و ورودی ها را در آن فیت میکنیم. متغیر all\_error میانگین خطا ها را در هر مرحله آموزش نشان میدهد. نهایتا نمودار یانگین خطا بر حسب تعداد دفعات آموزش را رسم میکنیم. مشاهده میشود که این میانگین به مقدار ثابتی میل میکند:**



**سوال2)**

**ابتدا کتابخانه های لازم را ایمپورت میکنیم:**

import numpy as np

from sklearn.datasets import make\_blobs

from matplotlib import pyplot as plt

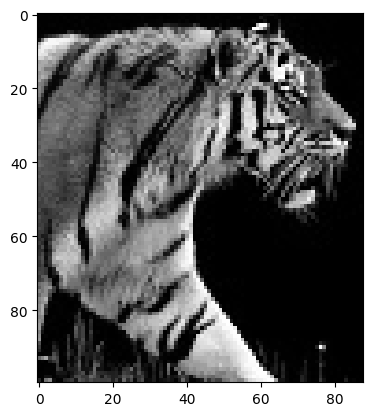
import cv2

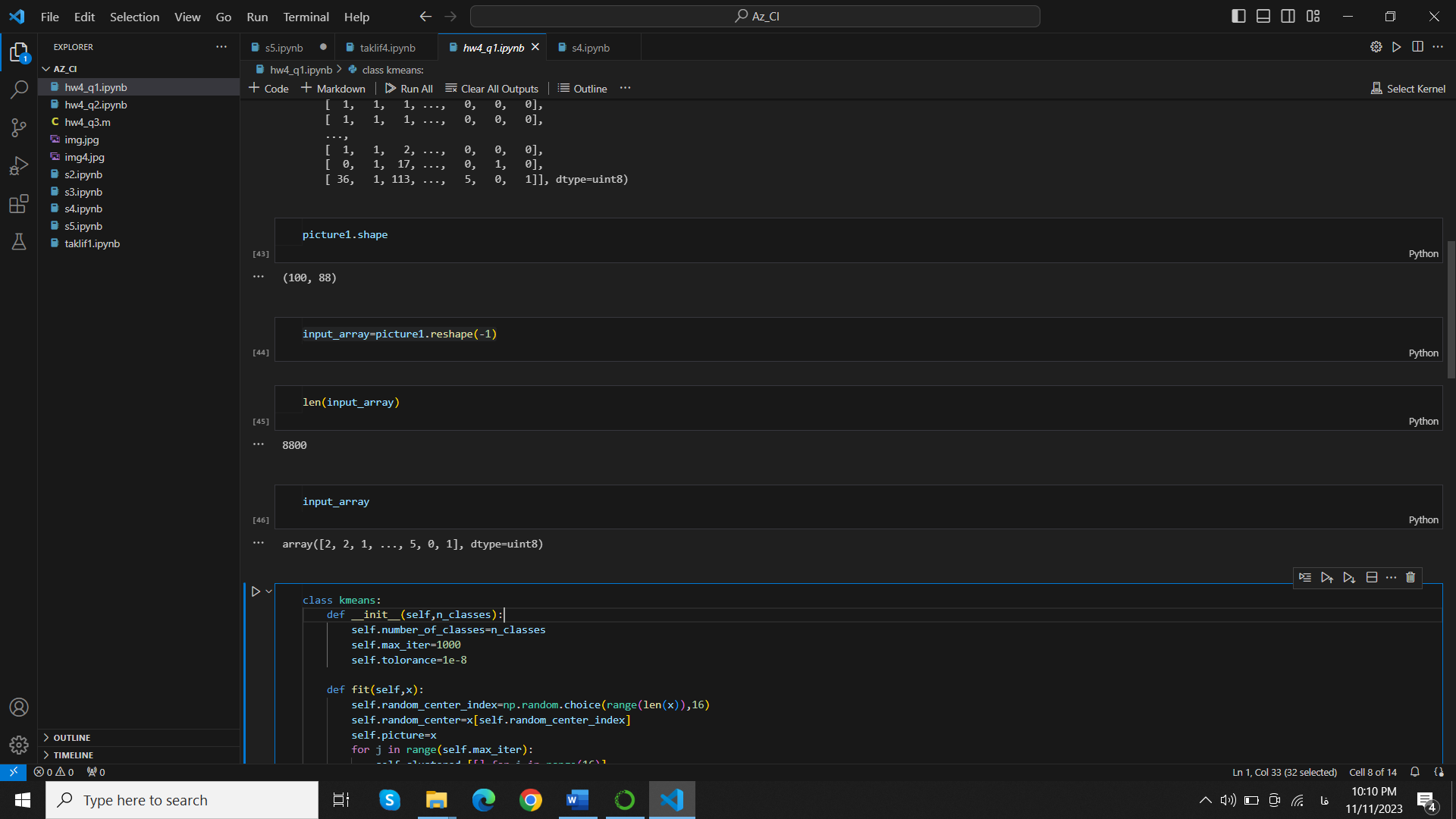
**با استفاده از کتابخانه opencv عکس را خوانده و سیاه سفید میکنیم و نتیجه را میبینیم:**

picture1 = cv2.imread("./img4.jpg")

picture1 = cv2.cvtColor(picture1, cv2.COLOR\_BGR2GRAY)

plt.imshow(picture1, cmap="gray")

****



**ایعدا عکس را بررسی کرده و آن را به آرایه ی تک بعدی تبدیل میکنیم. نهایتا تعداد درایه های آرایه را بررسی میکنیم:**

**تا اینجا عملیات preprocess انجام شد. حال باید مدل را طراحی کنیم.**

**کلاس kmeans را ایجاد کرده و در متد init، متغیر های تعداد کلاس و تعداد دفعات ترین و خطا را تعریف میکنیم:**

class kmeans:

    def \_\_init\_\_(self,n\_classes):

        self.number\_of\_classes=n\_classes

        self.max\_iter=1000

        self.tolorance=1e-8

**متد بعدی، fit است که توضیح آن به شرح زیر است:**

**از بین داده های ورودی، 16 داده بصورت رندوم، بعنوان مرکز خوشه انتخاب میشوند.**

**همچنین داده ها را در متغیر دیگری به اسم picture میریزیم. حال در حلقه تکرار آموزش، متغیر cluster را که لیستی شامل 16 لیست خالی است را میسازیم.**

**سپس در حلقه دیگری، درایه هارا در لیستی میریزیم که نزدیک ترین فاصله به مرکز خوشه آن را دارد. نهایتا میانگین اعضای هر لیست را بعنوان مرکز دسته جدید در نظر میگیریم و این کار تکرار میشود.**

**در مرحله آخر، درایه های متغیر picture را با نزدیک ترین مرکز خوشه جایگزین میکنیم.**

def fit(self,x):

        self.random\_center\_index=np.random.choice(range(len(x)),16)

        self.random\_center=x[self.random\_center\_index]

        self.picture=x

        for j in range(self.max\_iter):

            self.clustered=[[] for i in range(16)]

            for i in range(8800):

                distance=np.abs(self.random\_center-x[i])

                minimum\_index=np.argmin(distance)

                self.clustered[minimum\_index].append(x[i])

            for i in range(16):

                self.random\_center[i]=np.mean(self.clustered[i], axis=0)

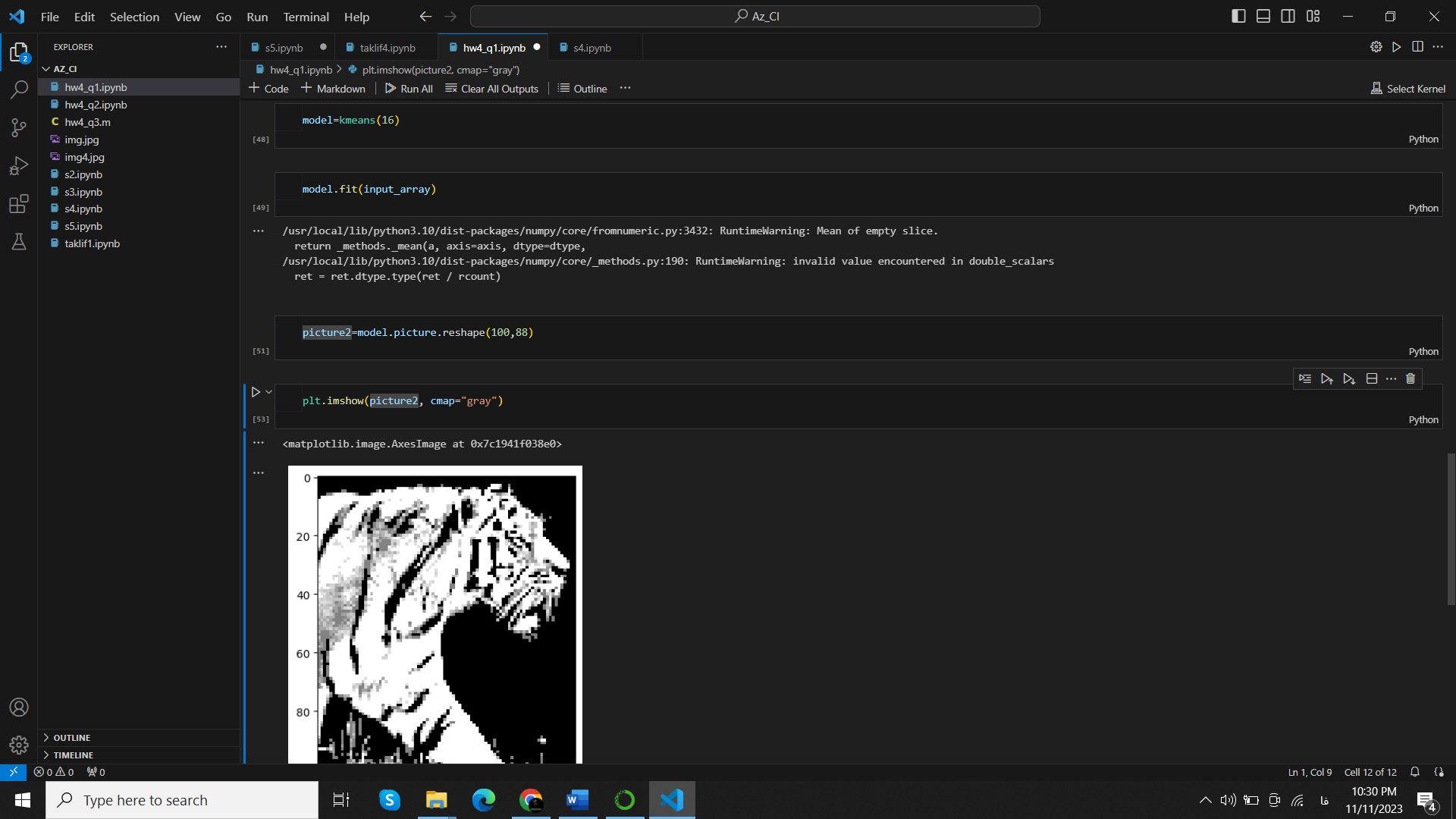
        for i in range(8800):

            distance=np.abs(self.random\_center-x[i]\*np.ones(16))

            minimum\_index=np.argmin(distance)

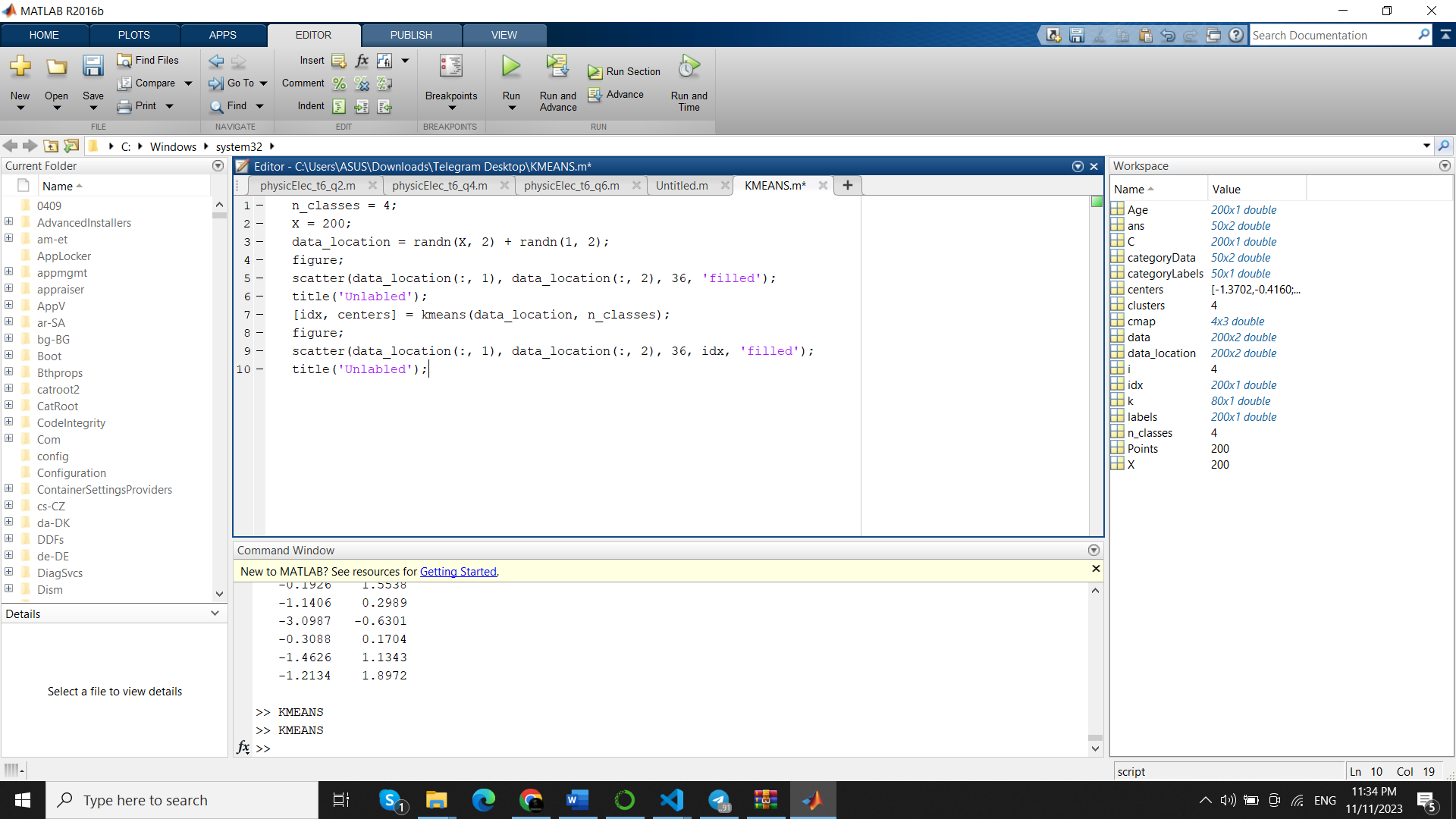
            self.picture[i]=self.random\_center[minimum\_index]

**حال از مدل استفاده میکنیم. متغیر picture از مدل را درون متغیر picture2 میریزیم و ابعاد آن را به ابعاد عکس بر میگردانیم و نتیجه را مشاهده میکنیم:**



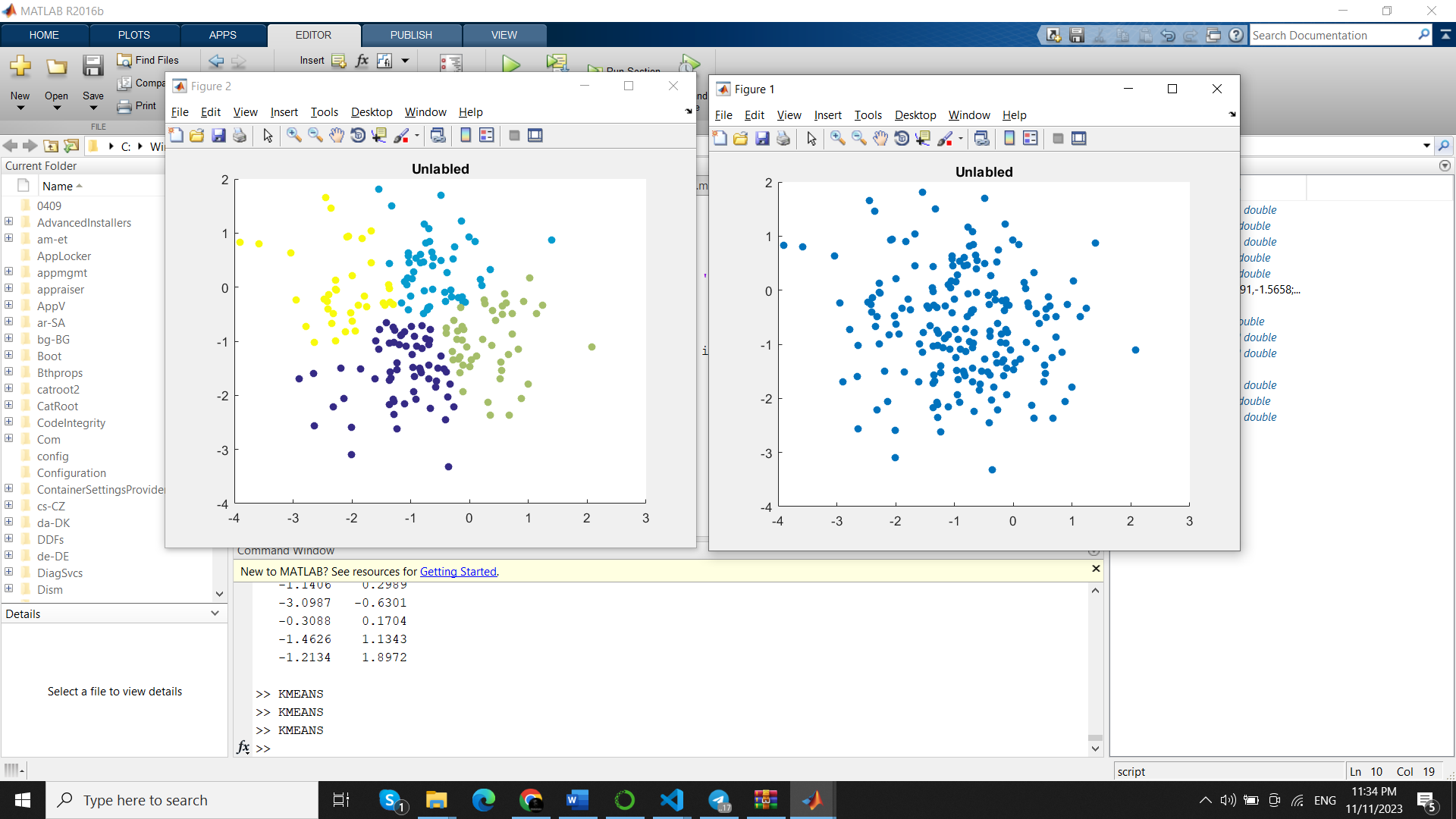
**سوال3)**

**کد پیاده شده در سوال 1 در متلب درتصویر زیر آورده شده است:**

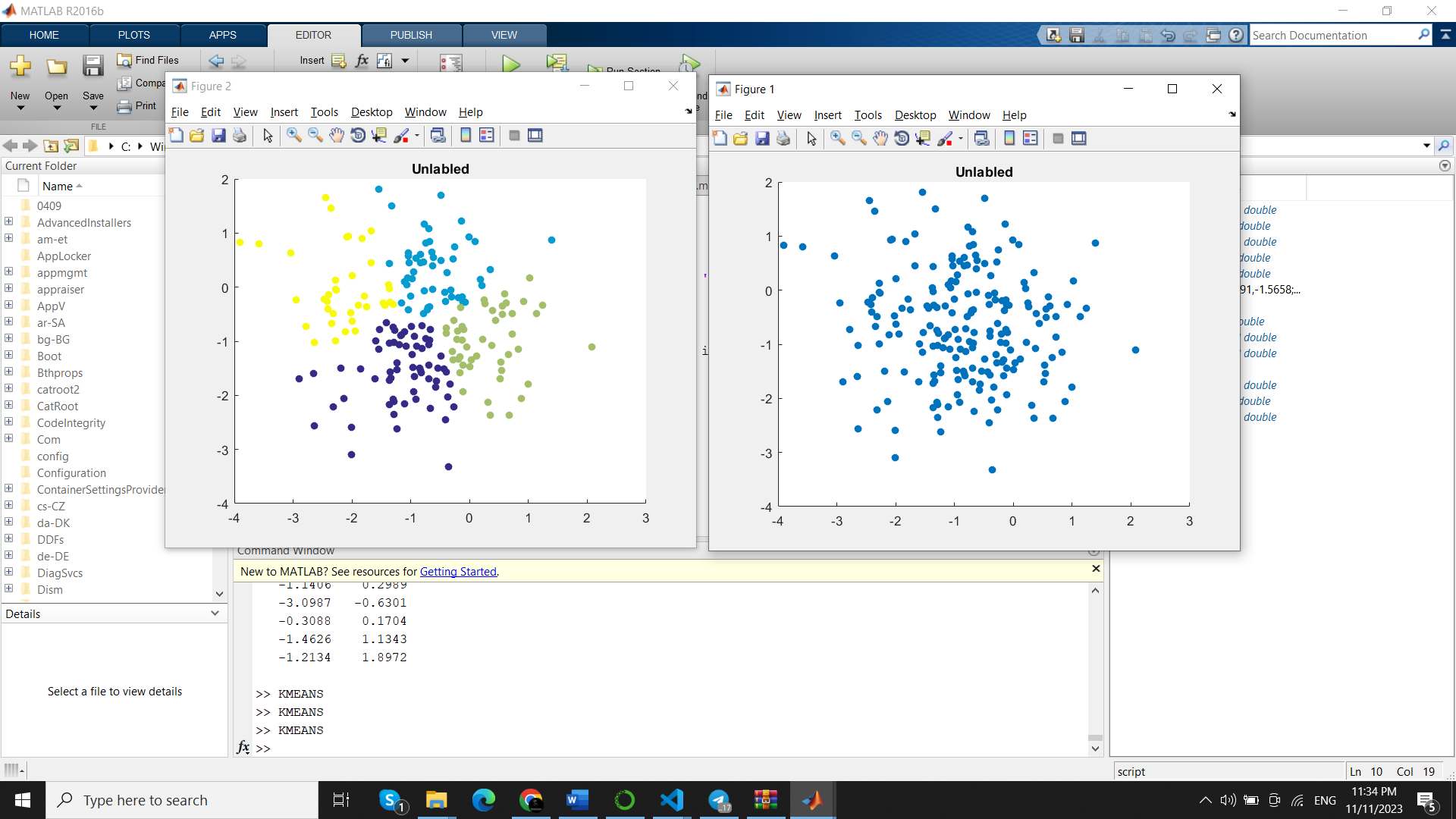


**تعداد دسته ها همانند سوال1، 4 دسته انتخاب شده است. همچنین 200 نقطه را بررشس خواهیم کرد. موقعیت نقاط را در متغیر data\_location بصورت رندوم انتخاب میکنیم.**

**حال ابتدا نمودار نقاط بدون لیبل را میبینیم:**



**سپس بکمک تابع kmeans در متلب، داده هارا دسته بندی کرده و نتیجه را مجددا میبینیم:**



**سوال4)**

**ابتدا باید اطلاعاتی درمورد شناسه مشتریان، سن، جنسیت و درآمد سالانه آنها از دیتاست به دست آورد . سپس باید آنها را بر اساس هرکدام از فاکتورها دسته بندی کنیم تا یک لیست دقیق برای ما به دست بیاید و بتوانیم راحت تر نمره خرید را دسته بندی کنیم.**

**پس از دسته بندی تمام فاکتور ها براساس سن، جنسیت و نمره خرید باید الگوریتم kmeans را پیاده سازی کنیم تا متوجه شویم برای چه مشتریانی باید تبلیغ کرد تا فروش بیشتری داشت.**

**میتوان از کتابخانه ها و دستور های موجود در Kaggle استفاده کرد و الگوریتم خرید را باید پیاده سازی کرد.  
حائز اهمیت است باید از کتابخانه sklearn برای پیاده سازی الگوریتم استفاده کنیم.**

**دستور های لازم عبارت است از:**

**Plt.scatter()  
kmeans.fit()**

**kmeans.fit\_predict()  
grid,plot,xticks  
add\_subplot() و ...  
مراحل زیر برای دسته بندی باید انجام شوند:**

**1. Data Preparation**

**2. Exploring the content of variables**

* 2.1 Countries
* 2.2 Customers and products
  + 2.2.1 Cancelling orders
  + 2.2.2 StockCode
  + 2.2.3 Basket price

**3. Insight on product categories**

* 3.1 Product description
* 3.2 Defining product categories
  + 3.2.1 Data encoding
  + 3.2.2 Clusters of products
  + 3.2.3 Characterizing the content of clusters

**4. Customer categories**

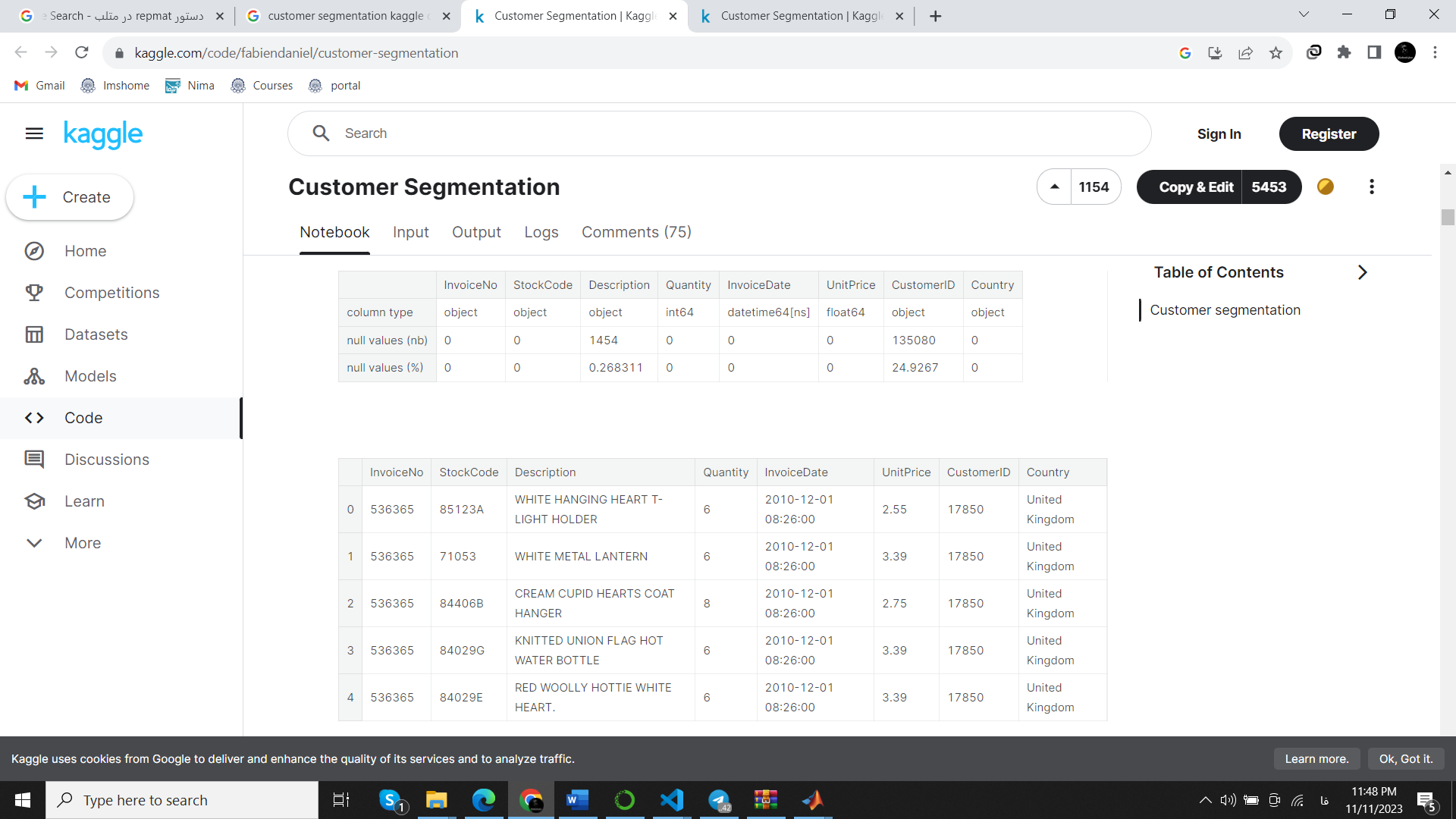
* 4.1 Formating data
  + 4.1.1 Grouping products
  + 4.1.2 Time spliting of the dataset
  + 4.1.3 Grouping orders
* 4.2 Creating customer categories
  + 4.2.1 Data enconding
  + 4.2.2 Creating categories

**5. Classifying customers**

* 5.1 Support Vector Machine Classifier (SVC)
  + 5.1.1 Confusion matrix
  + 5.1.2 Leraning curves
* 5.2 Logistic regression
* 5.3 k-Nearest Neighbors
* 5.4 Decision Tree
* 5.5 Random Forest
* 5.6 AdaBoost
* 5.7 Gradient Boosting Classifier
* 5.8 Let's vote !

**6. Testing the predictions**

**7. Conclusion**



**مرجع: سایت kaggle**