# 实验三

## 一、实验目的

1、掌握支持K-means的应用场景。

2、理解K-means的评估标准。

3、熟练scikit-learn、numpy、pandas等库的使用。

## 二、实验内容

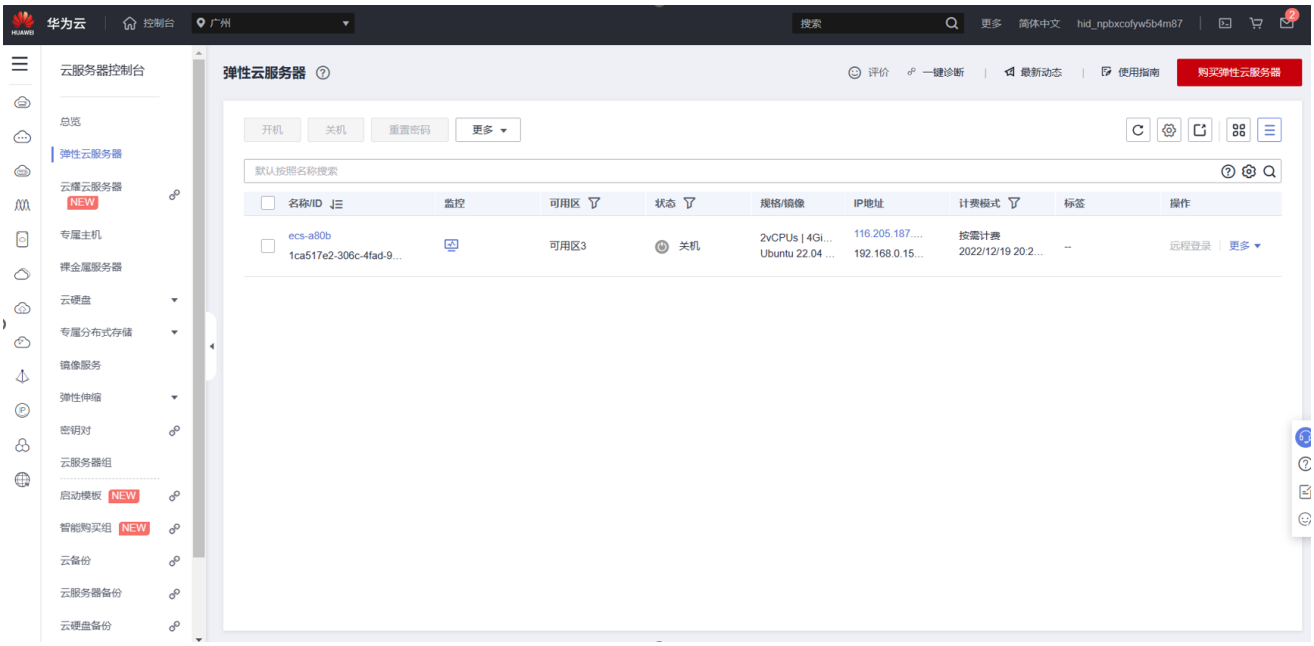
Iris数据集是常用的分类实验数据集，由Fisher于1936年收集整理。Iris也称鸢尾花卉数据集，是一类多重变量分析的数据集。数据集包含150个数据样本，分为3类，每类50个数据，每个数据包含4个属性。可通过花萼长度（Sepal.Length），花萼宽度（Sepal.Width），花瓣长度（Petal.Length），花瓣宽度（Petal.Width）4个属性预测鸢尾花卉属于Setosa（山鸢尾），Versicolour（杂色鸢尾），Virginica（维吉尼亚鸢尾）三个种类中的哪一类。更多数据集介绍可参考：https://archive.ics.uci.edu/ml/datasets/Iris/

## 三、数据预处理

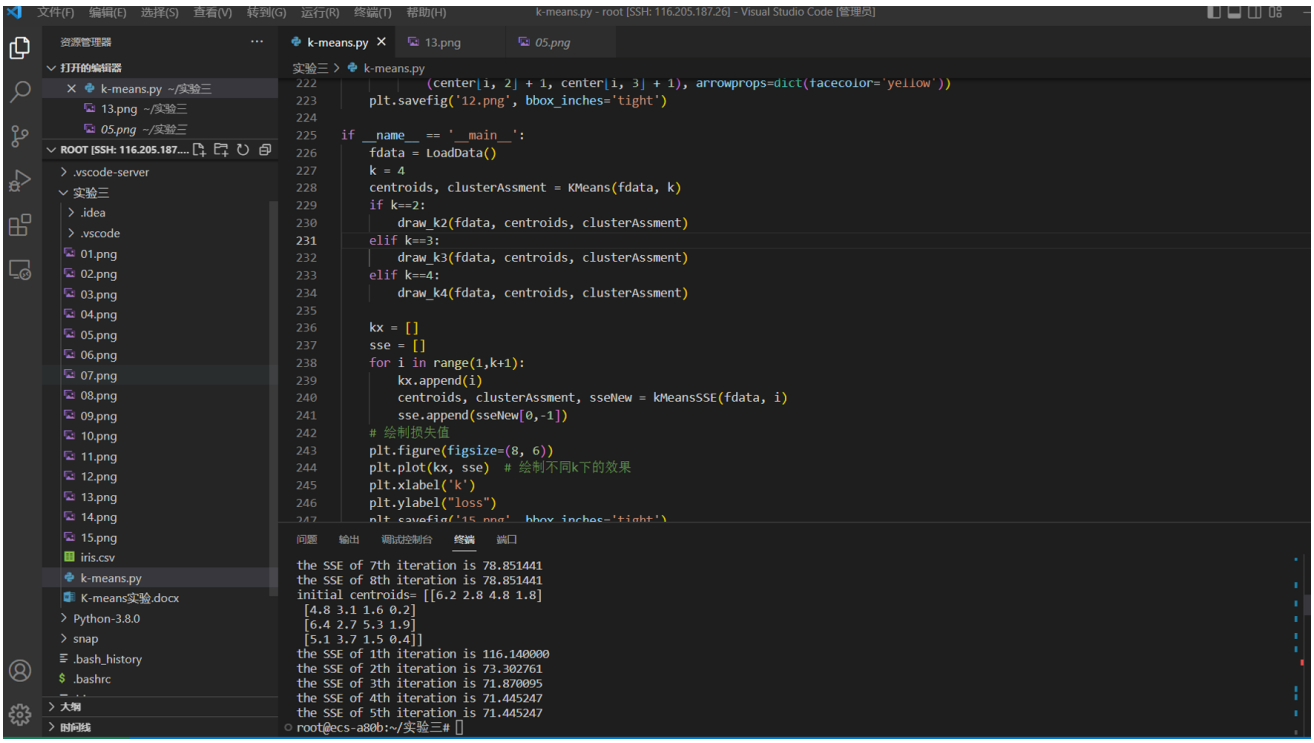
```python
def LoadData():
    data = pd.read_csv("iris.csv")
    # 获取所有列，并存入一个数组中
    ori_data = np.array(data)
    # 删去数组中的序号列和分类结果列
    delete_index = [0,5]
    fdata = np.delete(ori_data,delete_index,axis=1)
    # 返回最后处理好的数据集
    return fdata
```

# 四、手写K-means算法

## 1、华为云部署截图




## 2、在云服务器上运行




## 3、手写K-means算法实现代码

```
# k均值聚类算法
def KMeans(dataSet, k):
    m = np.shape(dataSet)[0]  # 行数150
    # 第一列存每个样本属于哪一簇(四个簇)
    # 第二列存每个样本的到簇的中心点的误差
    clusterAssment = np.mat(np.zeros((m, 2)))  #
.mat()创建150*2的矩阵
    clusterChange = True
    # 1.初始化质心centroids
    centroids = randCent(dataSet, k)  # 4*4
    while clusterChange:
        # 样本所属簇不再更新时停止迭代
        clusterChange = False
        # 遍历所有的样本（行数150）
        for i in range(m):
            minDist = 100000.0
            minIndex = -1
            # 遍历所有的质心
            # 2.找出最近的质心
```

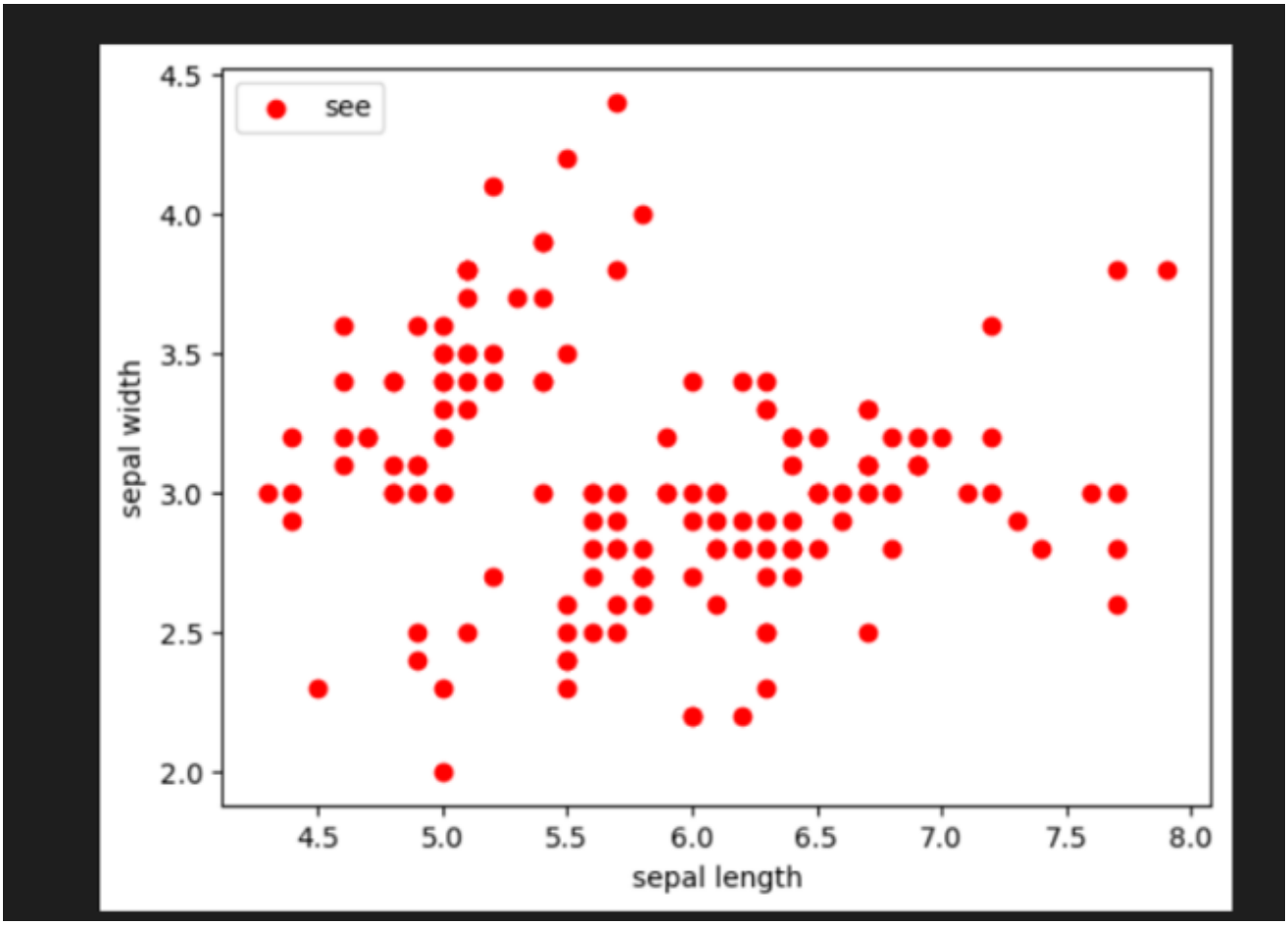
```
            for j in range(k):
                # 计算该样本到4个质心的欧式距离，找到距离最近的那个质心minIndex
                distance = distEclud(centroids[j, :], dataSet[i, :])
                if distance < minDist:
                    minDist = distance
                    minIndex = j
            # 3.更新该行样本所属的簇
            if clusterAssment[i, 0] != minIndex:
                clusterChange = True
                clusterAssment[i, :] = minIndex, minDist ** 2
        # 4.更新质心
        for j in range(k):
            # np.nonzero(x)返回值不为零的元素的下标，它的返回值是一个长度为x.ndim(x的轴数)的元组
            # 元组的每个元素都是一个整数数组，其值为非零元素的下标在对应轴上的值。
            # 矩阵名.A 代表将 矩阵转化为array数组类型

            # 这里取矩阵clusterAssment所有行的第一列，转为一个array数组，与j（簇类标签值）比较，返回true or false
            # 通过np.nonzero产生一个array，其中是对应簇类所有的点的下标值（x个）
            # 再用这些下标值求出dataSet数据集中的对应行，保存为pointsInCluster（x*4）
            pointsInCluster = dataSet[np.nonzero(clusterAssment[:, 0].A == j)[0]]  # 获取对应簇类所有的点（x*4）
            centroids[j, :] = np.mean(pointsInCluster, axis=0)  # 求均值，产生新的质心
```

```
            # axis=0，那么输出是1行4列，求的是
pointsInCluster每一列的平均值，即axis是几，那就表明哪一维度被
压缩成1
    print("cluster complete")
    return centroids, clusterAssment

def kMeansSSE(dataSet,k):
    m = np.shape(dataSet)[0]
    #分配样本到最近的簇：存[簇序号,距离的平方]
    clusterAssment=np.mat(np.zeros((m,2)))
    #step1:#初始化聚类中心
    centroids = randCent(dataSet, k)
    print('initial centroids=',centroids)
    sseOld=0
    sseNew=np.inf
    iterTime=0 #查看迭代次数
    while(abs(sseNew-sseOld)>0.0001):
        sseOld=sseNew
        #step2:将样本分配到最近的质心对应的簇中
        for i in range(m):
            minDist=100000.0;minIndex=-1
            for j in range(k):
                #计算第i个样本与第j个质心之间的距离

distJI=distEclud(centroids[j,:],dataSet[i,:])
                #获取到第i样本最近的质心的距离,及对应簇序号
                if distJI<minDist:
                    minDist=distJI;minIndex=j
            clusterAssment[i,:]=minIndex,minDist**2 #
分配样本到最近的簇
        iterTime+=1
        sseNew=sum(clusterAssment[:,1])
```

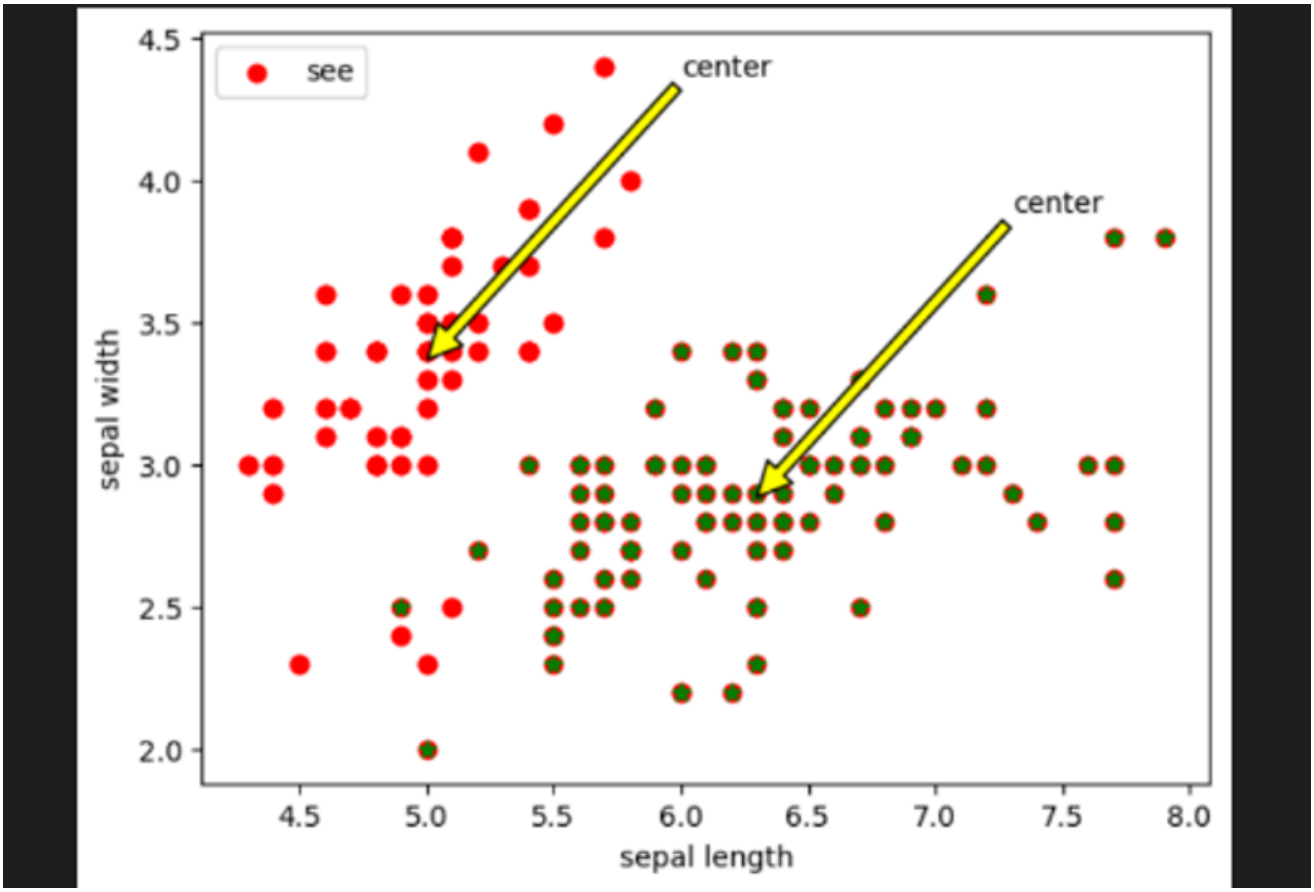
```
        print('the SSE of %d'%iterTime + 'th iteration
is %f'%sseNew)
        #step3:更新聚类中心
        for cent in range(k):
            #样本分配结束后，重新计算聚类中心

ptsInClust=dataSet[np.nonzero(clusterAssment[:,0].A==c
ent)[0]]
            #按列取平均,mean()对array类型
            centroids[cent,:] = np.mean(ptsInClust,
axis=0)
    return centroids, clusterAssment,sseNew
```

## 五、实验结果可视化

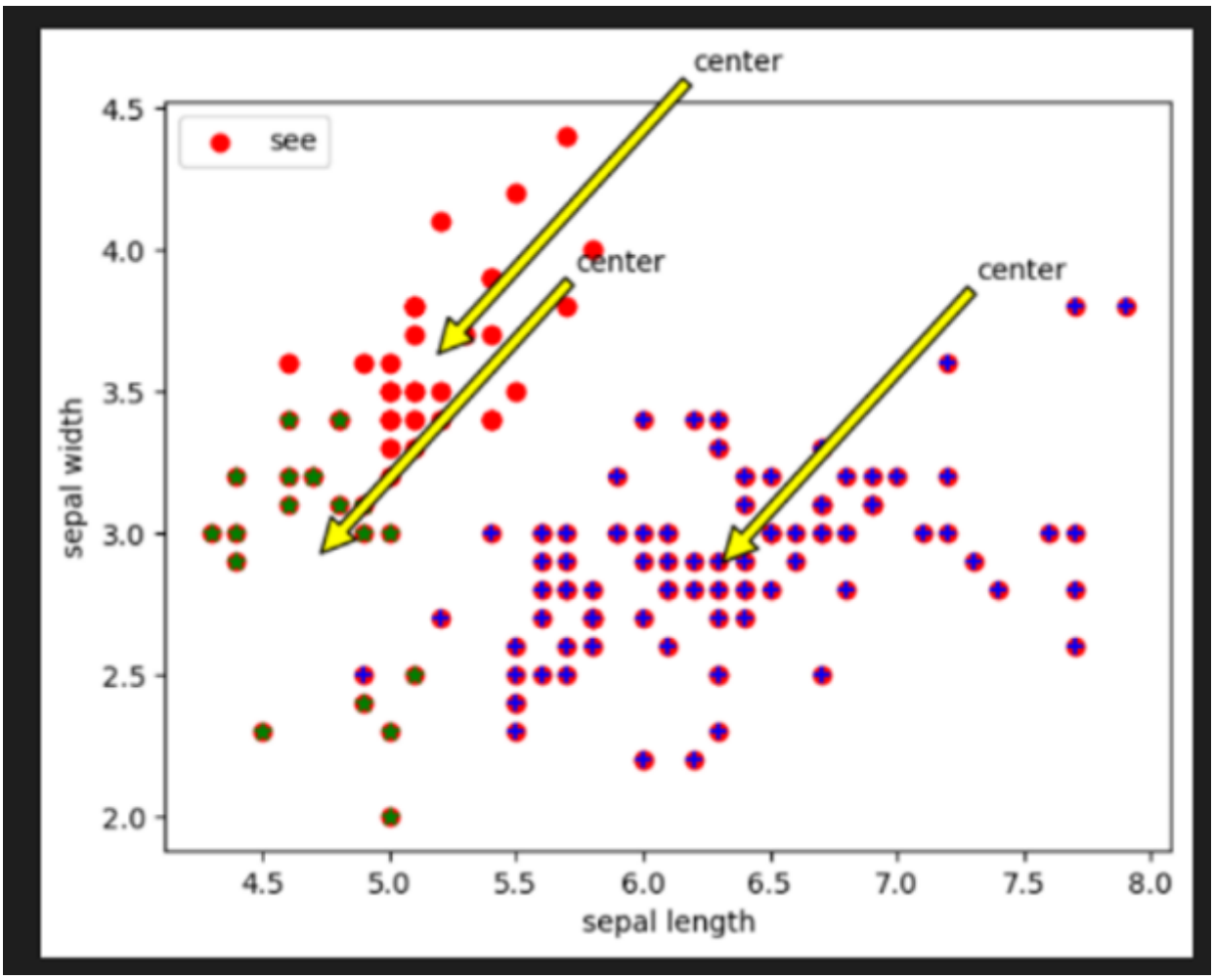
1、前两个维度情况下原始图



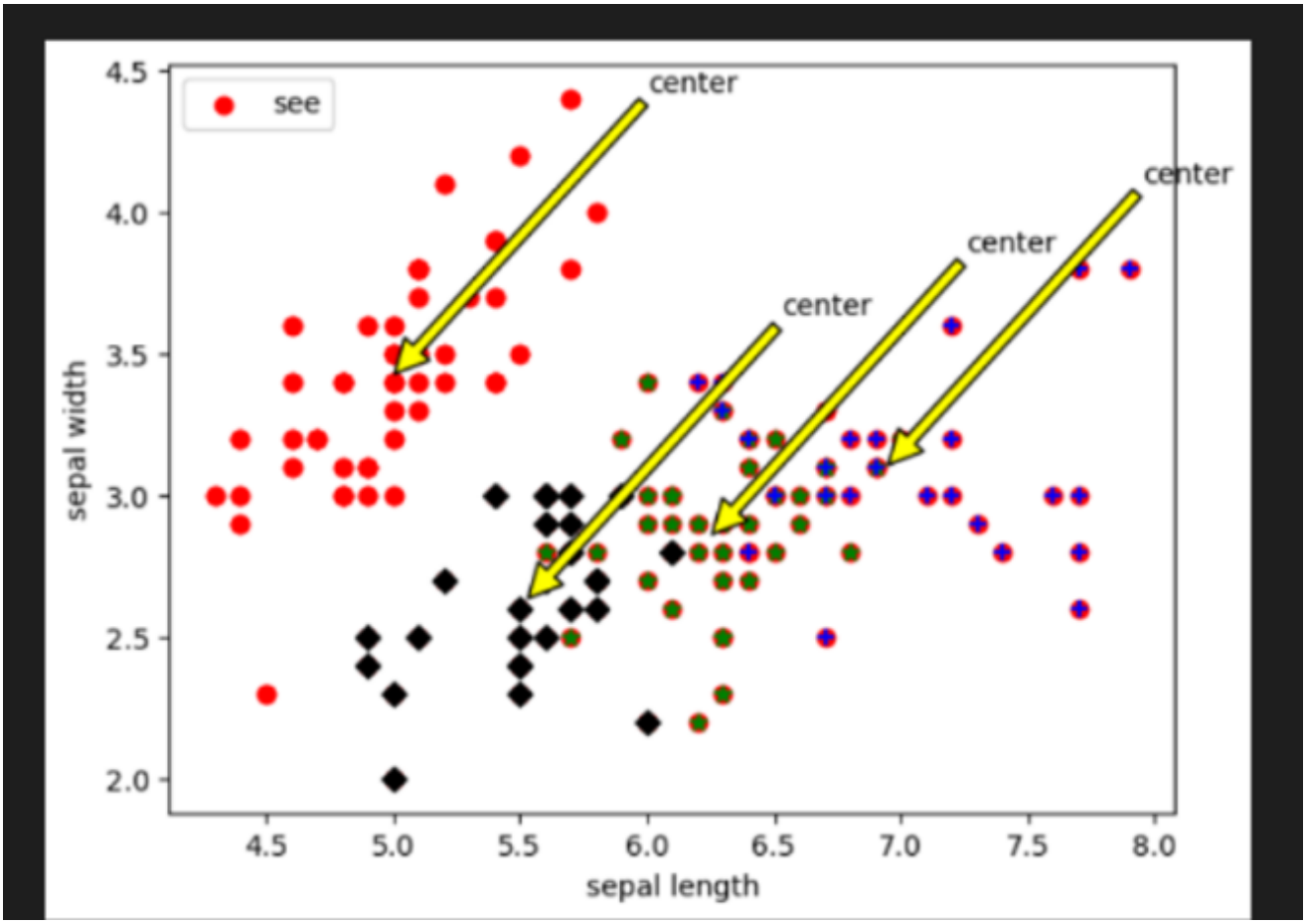
2、前两个维度下，k=2聚类结果



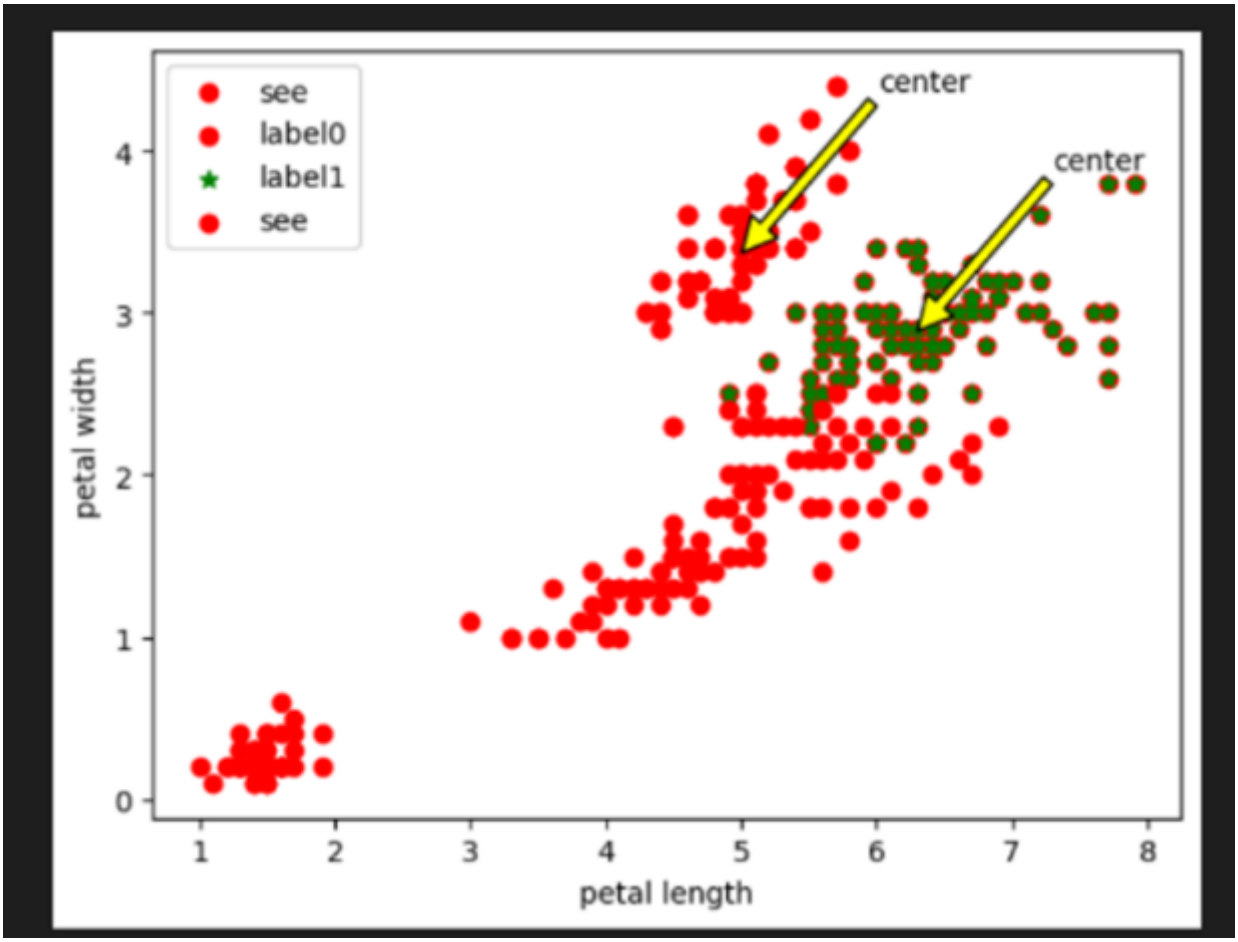

3、前两个维度下，k=3聚类结果



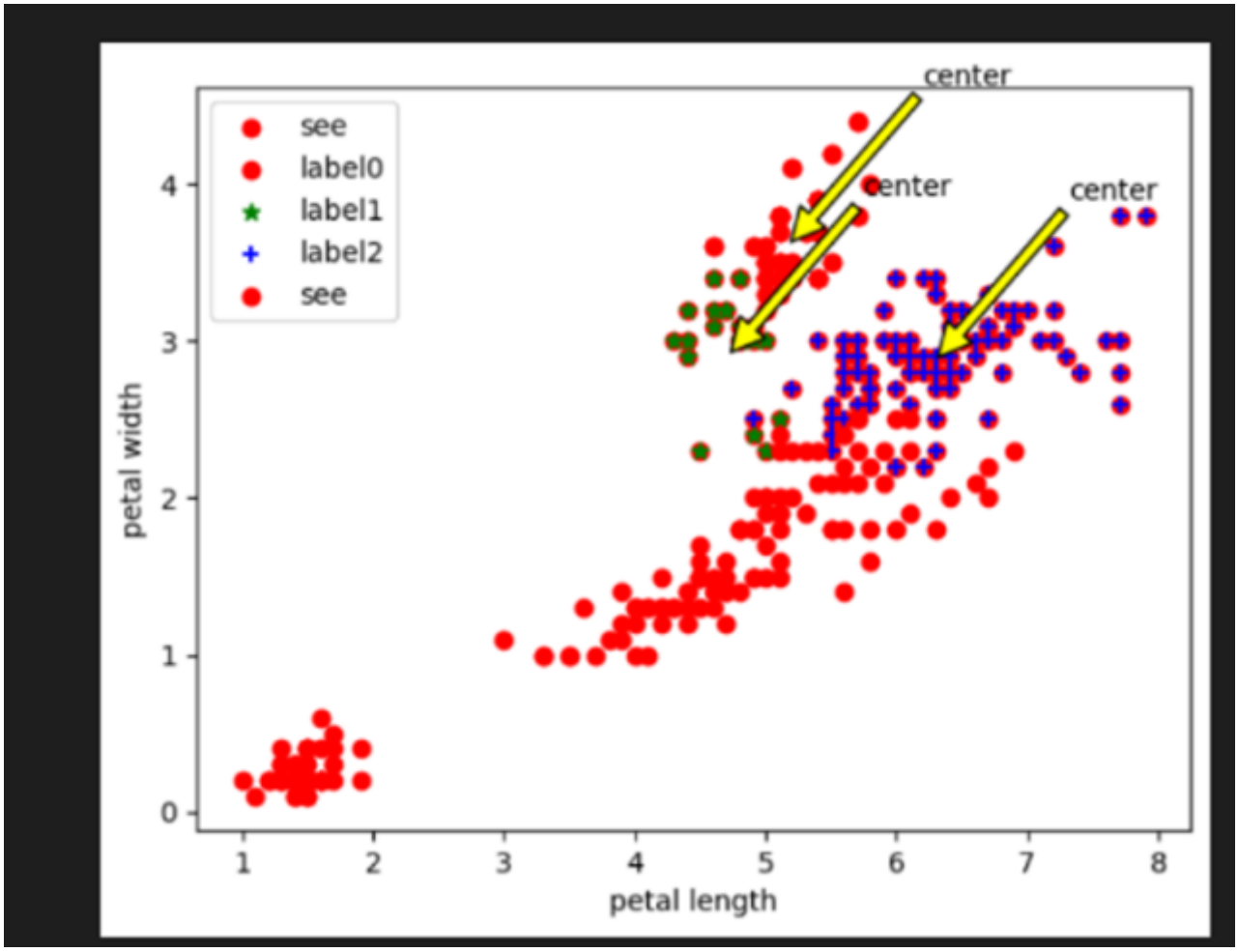

4、前两个维度下，k=4聚类结果



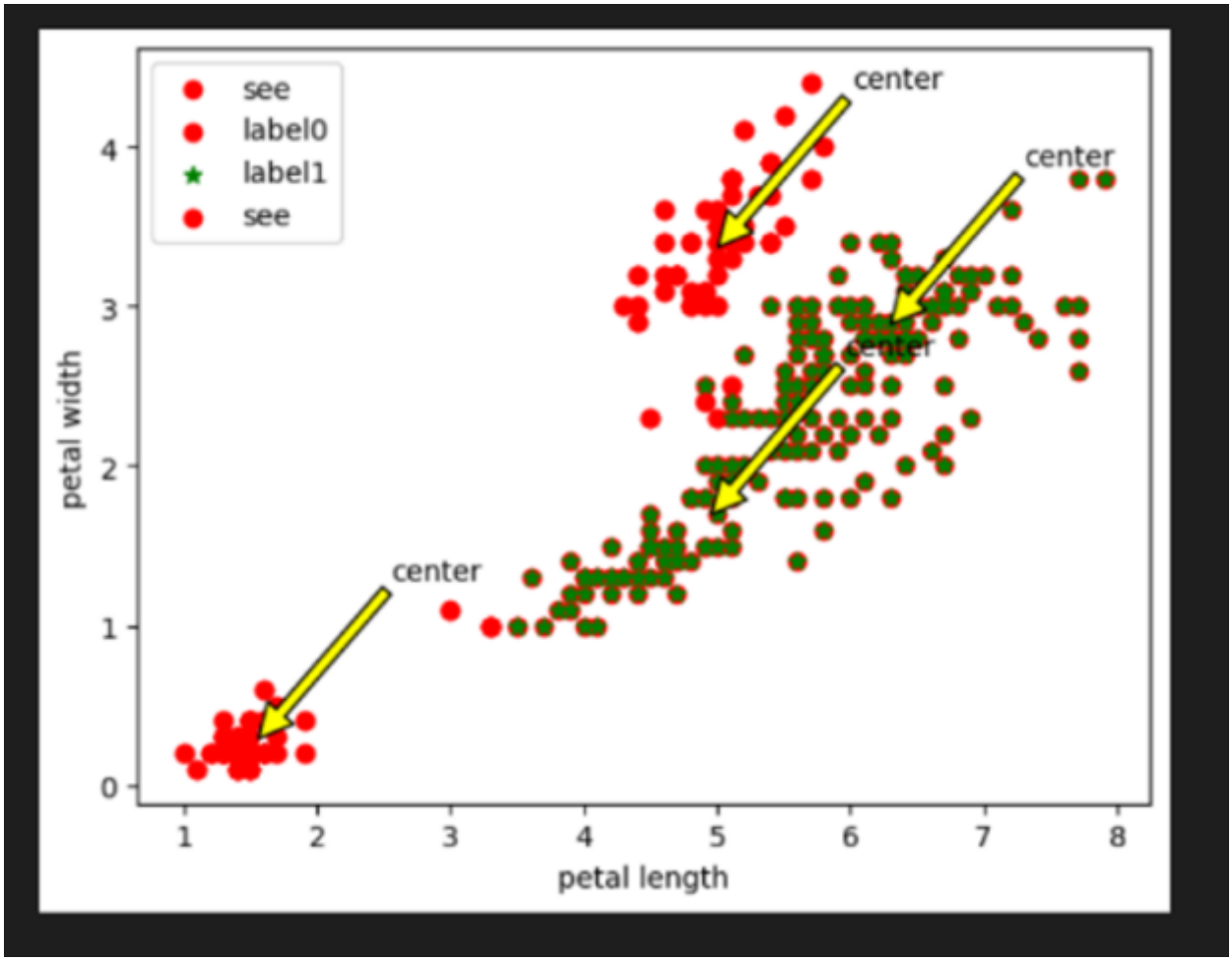
5、后两个维度下，k=2聚类结果



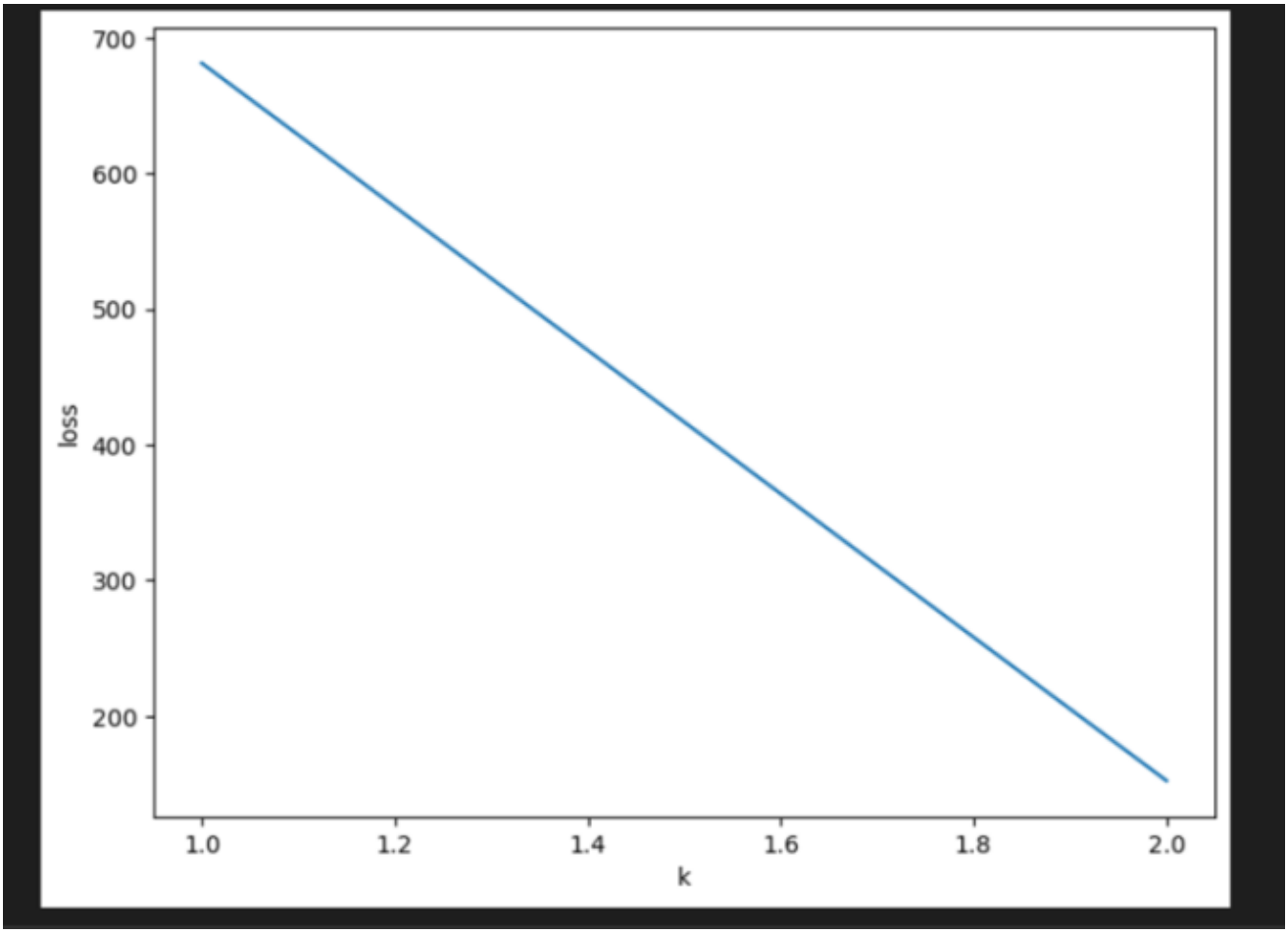

6、后两个维度下，k=3聚类结果



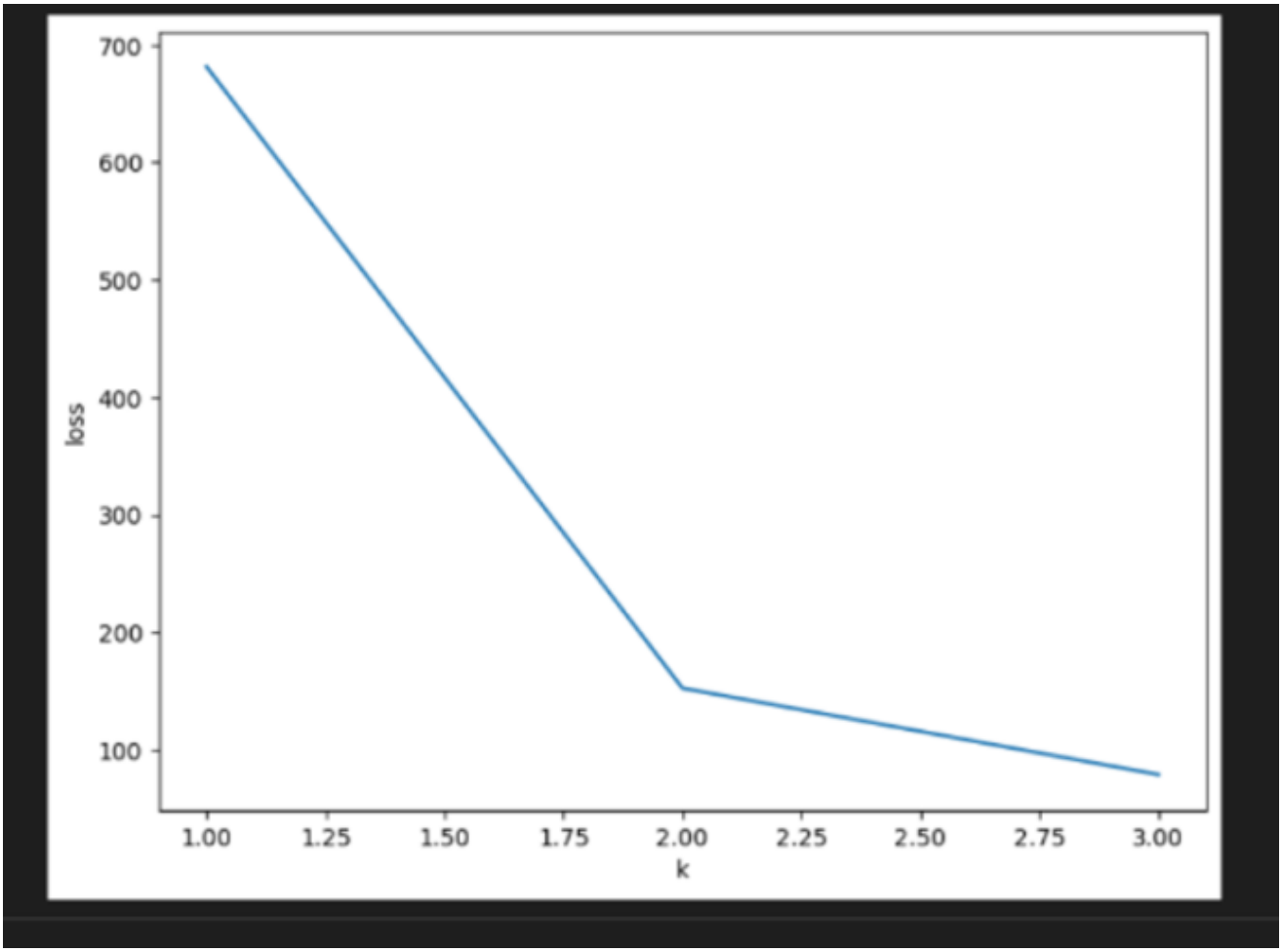

7、后两个维度下，k=4聚类结果


see
label0
label1
see
center
center
center
center
petal width
petal length
0
1
2
3
4
1
2
3
4
5
6
7
8

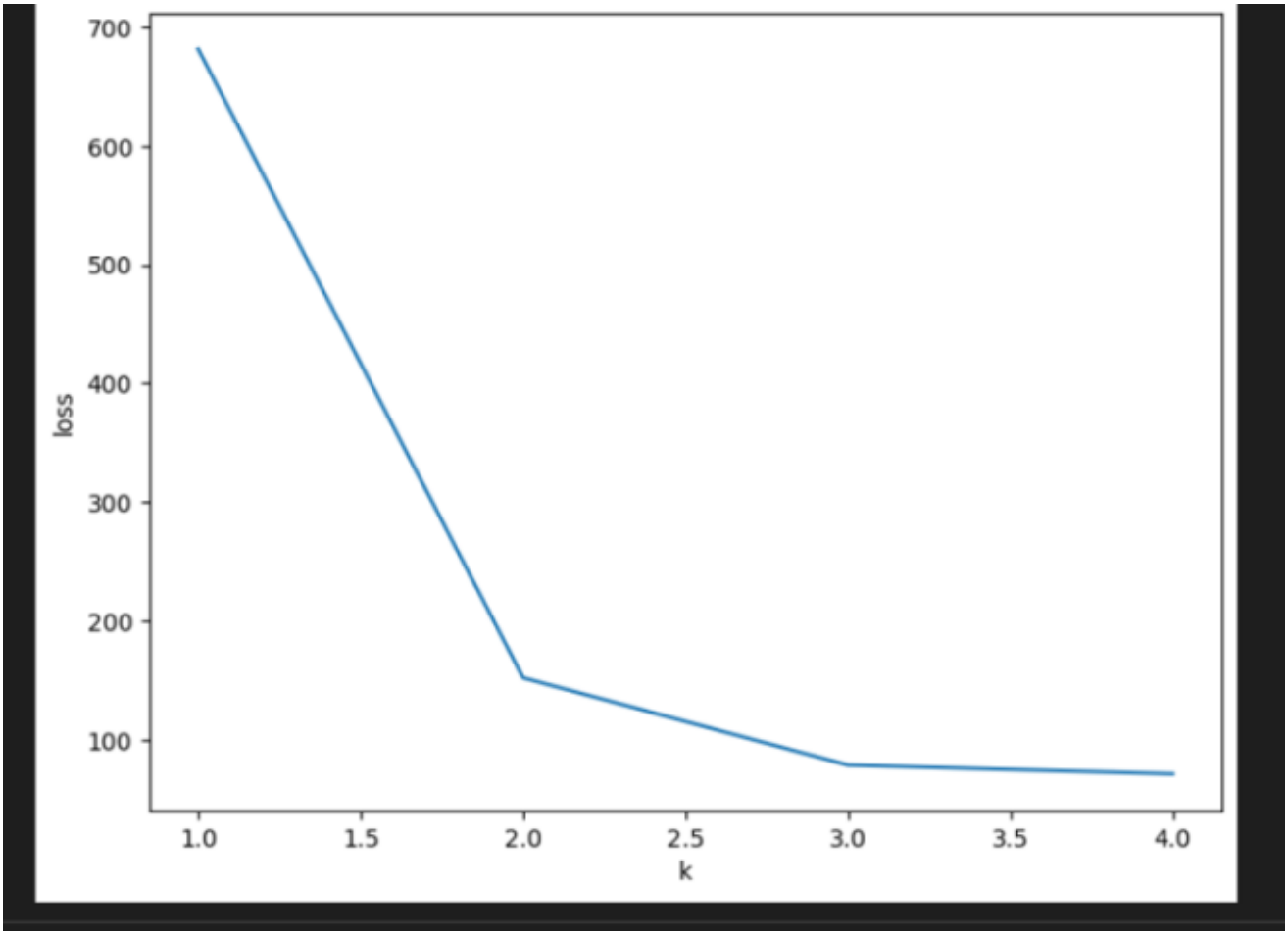
8、k=2时手肘图


loss
k
700
600
500
400
300
200
1.0
1.2
1.4
1.6
1.8
2.0

9、k=3时手肘图




10、k=4时手肘图




## 六、分析实验结果

由手肘图分析可得，当k=3时，曲率最大，故当k=3时聚类结果最好，故最佳的k值为3。同时，在k=3时的聚类图与另外两个k值下的聚类图作比较时，可以看出，k=3时的聚类效果更好，故最佳的k值为3。