Kmeans机器学习实验——171250574杨逸存

实验简介

- 本实验使用python语言编写模拟实现了Kmeans聚类算法
- 使用到的python库有：numpy、matplotlib

算法代码详解

算法原理

- Kmeans输入：
  - N个D维样本点{x1, x2, ...xn}
  - 拟定的聚类个数K
- 初始化：
  - 随机初始化K个不同的样本点作为初始聚类中心
- 迭代：
  - 对于每个样本点xi都将其指定为离其最近的聚类中心点簇，本实验中距离的定义为向量的欧式距离
  - 重新计算聚类中心，每个中心点坐标为所有属于这个簇的点的坐标的平均值
  - 迭代直至收敛，本实验中收敛定义为损失函数变化小于一个阈值eps
- 损失函数：每个样本点离所属簇中心点的距离之和

代码详解

- 主方法代码

  初始化100个点和聚簇中心后进入迭代，每次迭代先更新簇中心，再进行聚类判断，再计算损失函数值，最后记录可视化图信息。

```
if __name__ == '__main__':
    points = init_random_points(100)
    centroids = init_k_centroids(points, k=3)
    cluster_dict = cal_clusters(points, centroids);

    curr_loss = cal_loss(cluster_dict, centroids);
    prev_loss = float("inf")
    eps = 0.0001  # 迭代终止条件
    iter_time = 0;  # 迭代计数

    plt.figure(figsize=(15, 15))
    plot_clusters(cluster_dict, centroids, iter_time)

    while abs(curr_loss - prev_loss) > eps:
```

```
        # 迭代次数加一
        iter_time += 1
        # 更新上一次误差
        prev_loss = curr_loss
        # 更新中心点
        centroids = cal_centroids(cluster_dict)
        # 进行聚类
        cluster_dict = cal_clusters(points, centroids)
        # 计算误差
        curr_loss = cal_loss(cluster_dict, centroids)
        # 可视化过程
        plot_clusters(cluster_dict, centroids, iter_time)
```

- 核心方法代码
  - 计算每一次迭代的聚类结果

    使用一个字典cluster_dict记录聚类结果，key为簇中心点index（聚类id），value为属于这一个类簇的点的坐标列表。对于每个点，计算其距离哪一个簇中心最近，并归入此簇中心代表的簇。

```python
def cal_clusters(points, centroids) -> dict:
    '''
    计算每个点所属的簇
    :param points: 点集
    :param centroids: 中心点点集
    :return: (中心点，所属点列表)字典对象
    '''
    cluster_dict = dict()
    for point in points:
        distance = 0.0
        min_distance = float('inf')
        centroid_idx = -1
        for idx in range(len(centroids)):
            # 计算距离最近的中心点
            centroid = centroids[idx]
            distance = cal_distance(point, centroid)
            if(distance < min_distance):
                min_distance = distance
                centroid_idx = idx
        if centroid_idx not in cluster_dict.keys():
            cluster_dict[centroid_idx] = []
        cluster_dict[centroid_idx].append(point)
    return cluster_dict
```

  - 重新选择簇中心点

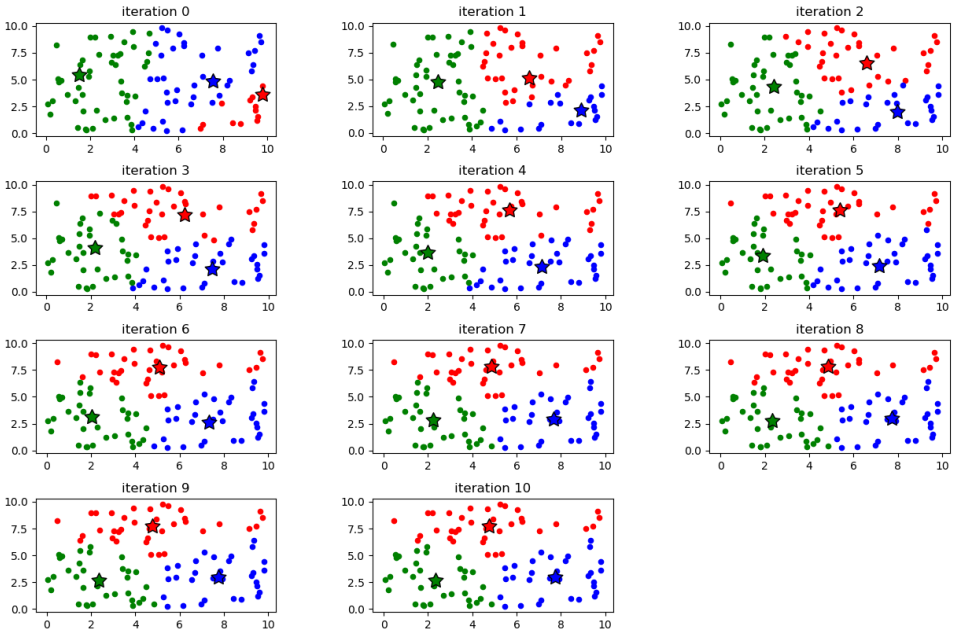
    新的簇中心点坐标为属于该簇的所有点坐标的平均值。

```python
def cal_centroids(cluster_dict) -> np.ndarray:
    '''
    重新计算中心点集
    :param cluster_dict: 当前聚簇结果
    :return:
    '''
    new_centroids = []
    for centroid_idx in cluster_dict.keys():
        this_cluster = cluster_dict[centroid_idx]
        new_centroid = np.mean(this_cluster, axis=0)
        new_centroids.append(new_centroid)
    return np.array(new_centroids)
```

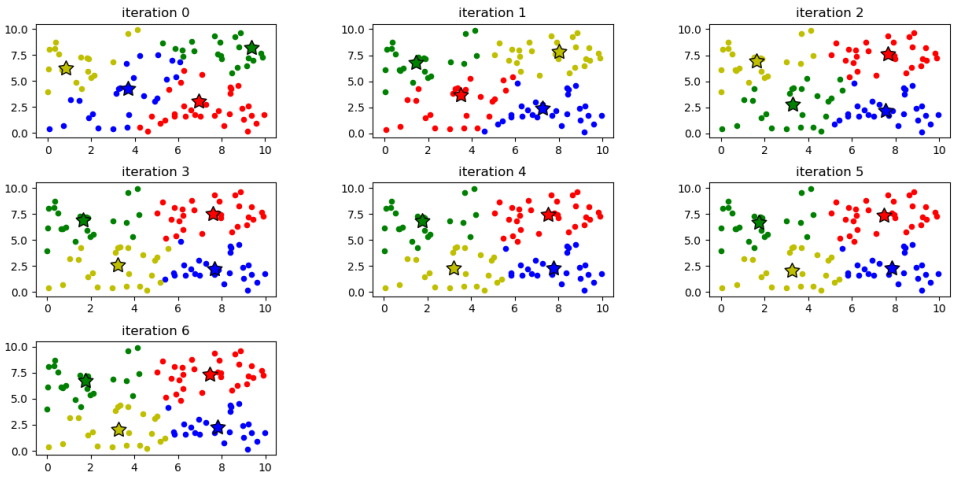
实验结果与发现

进行取K=3、4的实验

- K = 3



- K = 4



- 普通实心圆点为随机样本点，黑色边框星形图案为簇中心，颜色相同的点为同一簇的点。
- 根据迭代顺序展示出聚类结果和簇中心位置的变化情况，初期变化较大，迭代末期趋于稳定。
- 初始化簇中心会一定程度影响最终聚类结果。并且初始化簇中心距离过于接近会导致迭代初期聚类不平衡的现象。但最终各类间距大，类内聚性高，效果很好。