一、 实验过程

1. 数据集及预处理

聚类数据和分类采用同一组数据，数据前期处理一样，参见分类算法数据处理，这里补充其他的处理方法。

- 数据归一化处理

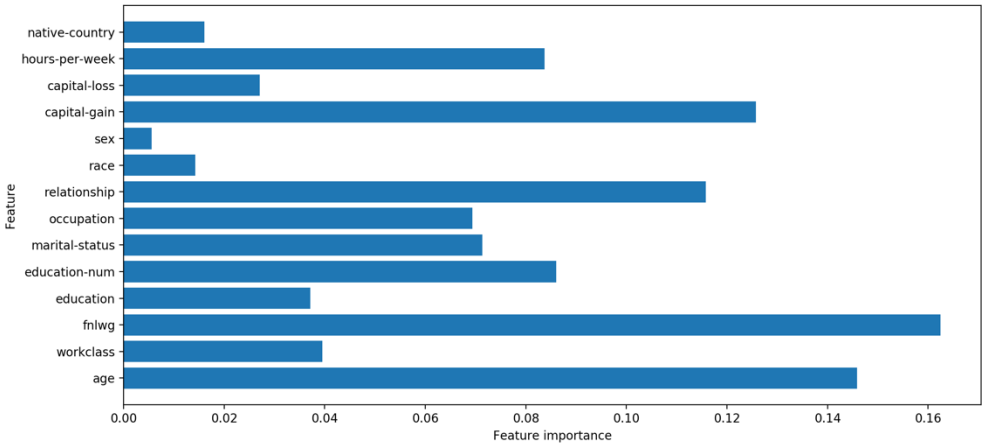
附录中原始数据的属性不一样，会使得聚类收敛慢、训练时间长。数据范围大的输入在模式分类中的作用可能会偏大，而数据范围小的输入作用就可能会偏小，需要进行数据归一化的处理。

```
#数据集
loan = data[['fnlwg','hours-per-week','marital-status','occupation','age','capital-gain','relationship','education-num']]
 # 数据标准化
data_zs = 1.0 * (loan - loan.mean()) / loan.std()  # 数据标准化
```

- 特征选择

利用随机森林来对特征进行选取，对比不同的特征组合对于模型的预测效果。

```
# 特征重要程度
print("特征重要度：\n{}".format(tree.feature_importances_))
adult_features=[x for i,x in enumerate(data.columns) if i !=14]
print(adult_features)
def plot_feature_importances_diabetes(model):
    plt.figure(figsize=(13,6))
    n_features=14
    plt.barh(range(n_features), model.feature_importances_,align='center')
    plt.yticks(np.arange(n_features), adult_features)
    plt.xlabel("Feature importance")
    plt.ylabel("Feature")
    plt.ylim(-1,n_features)
plot_feature_importances_diabetes(clf)
plt.show()
```



由图片观察，各个特征的重要程度排序可知，fnlwg、age、capital-gain、relationship、education-num、hours-per-week、marital-status、occupation 的相关度较高，因此选择这 8 个属性作为衡量特征，其余不作考虑。

```
1. loan = data[['fnlwg','hours-per-week','marital-
   status','occupation','age','capital-gain','relationship','education-num']]
2. data_zs = 1.0 * (loan - loan.mean()) / loan.std()  # 数据标准化
```

- 降维处理显示

由于原来数据集是一个具有 14 个特征的高维数据集，没有办法直接可视化结果，因此我们采用 PCA 降维的方法将原数据降低到 2 维，画出聚类后的二维数据图像

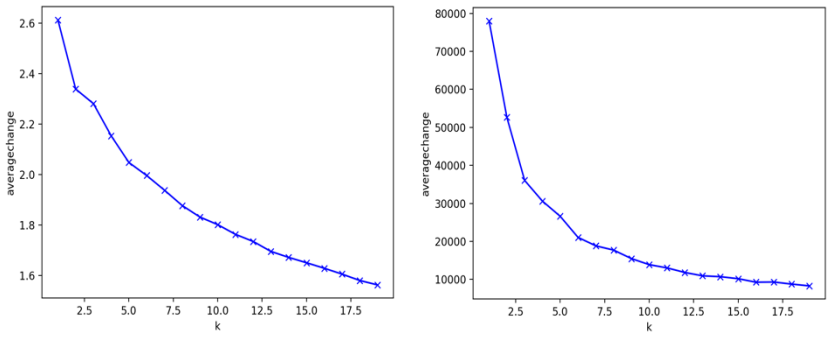
```
1. from sklearn.decomposition import PCA
2. pca = PCA(n_components=2)
3. pcadata = pd.DataFrame(pca.fit_transform(data_zs))
```

2. 模型构建

2.1kmeans

使用肘部法则，先判断聚类的数量。
肘部法则会把不同值的成本函数值画出来。随着值的增大，平均畸变程度会减小；每个类包含的样本数会减少，于是样本离其重心会更近。但是，随着值继续增大，平均畸变程度的改善效果会不断减低。值增大过程中，畸变程度的改善效果下降幅度最大的位置对应的值就是肘部。

```
from scipy.spatial.distance import cdist
K = range(1, 20)
meandistortions = []
for k in K:
    kmn = KMeans(n_clusters=k)
    kmn.fit(data_zs)
    meandistortions.append(sum(np.min(cdist(data_zs, kmn.cluster_centers_, '
euclidean'), axis=1)) / data_zs.shape[0])
plt.plot(K, meandistortions, 'bx-')
plt.xlabel('k')
plt.ylabel('averagechange')
plt.show()
```



从图中可以看出，值从 1 到 3 时，平均畸变程度变化最大。超过 3 以后，平均畸变程度变化显著降低。因此肘部就是 3，即为最佳的 K 值

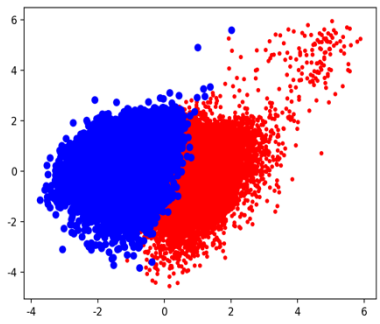
```
# kmeans 开始聚类
from sklearn.cr import KMeans
model.fit(data_zs)
# labels 为分类的标签
labels=model.labels_
#详细输出原始数据及其类别
r = pd.concat([loan, pd.Series(model.labels_, index = loan.index)], axis = 1
)
# 输出每个样本对应的类别
r.columns = list(loan.columns) + [u'聚类类别'] #重命名表头
# 降维显示聚类图像
```

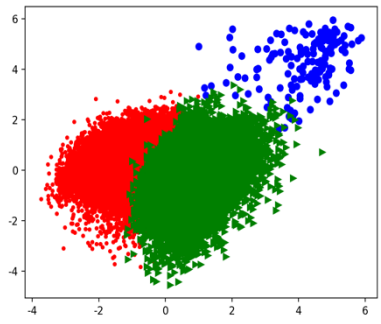
PCA 降维输出

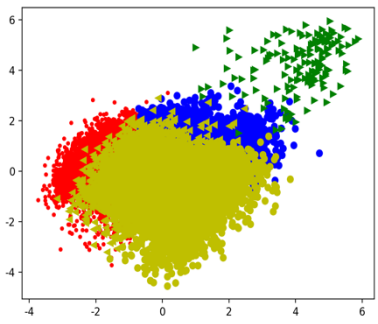
```
pca = PCA(n_components=2)
```

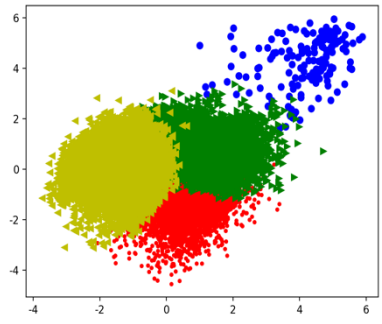
```
tsne = pd.DataFrame(pca.fit_transform(data_zs))
plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False #用来正常显示负号
# 不同类别用不同颜色和样式绘图
d = tsne[r[u'聚类类别'] == 0]
plt.plot(d[0], d[1], 'r.')
d = tsne[r[u'聚类类别'] == 1]
plt.plot(d[0], d[1], 'bo')
d = tsne[r[u'聚类类别'] == 2]
plt.plot(d[0], d[1], 'g')
d = tsne[r[u'聚类类别'] == 3]
plt.plot(d[0], d[1], 'y<')
plt.show()
```

下图分别是 n=2、3、4、5 时，kmeans 聚类的图像









CH 评估指标

Calinski-Harabasz(CH)指标：Calinski-Harabasz 分数值 ss 越大则聚类效果越好

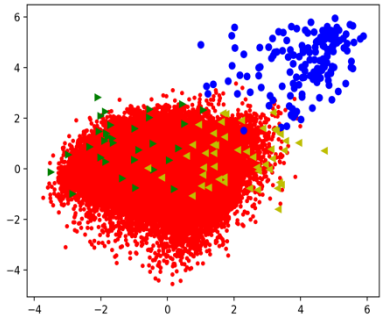
```
#  聚类评价 CH 指标
from sklearn import metrics
y_pred=model.predict(data_zs)
print(metrics.calinski_harabaz_score(data_zs, y_pred))
```

N_CLUSTERS	3	4	5	6
CH	6014	5538	5343	5009

2.2 Meanshift

```
# MeanShift 算法
from sklearn.datasets.samples_generator import make_blobs
from sklearn.cluster import MeanShift, estimate_bandwidth
bandwidth = estimate_bandwidth(loan, quantile=0.3, n_samples=None,random_state=0,   n_jobs=1)
# 设置均值偏移函数
ms = MeanShift(bandwidth=bandwidth, bin_seeding=True)
ms.fit(loan)
labels = ms.labels_
cluster_centers = ms.cluster_centers_
print("labels: ", labels)
n_clusters_ = len(np.unique(labels))
print(n_clusters_)
# 打印结果
# 详细输出原始数据及其类别
r = pd.concat([loan, pd.Series(ms.labels_, index = loan.index)], axis = 1)
#  输出每个样本对应的类别
r.columns = list(loan.columns) + [u'聚类类别'] #重命名表头
```

Meanshift 算法不需要提前设置聚类数目，算法结束后，将数据分为了 4 个类，降维可视化结果显示如下：



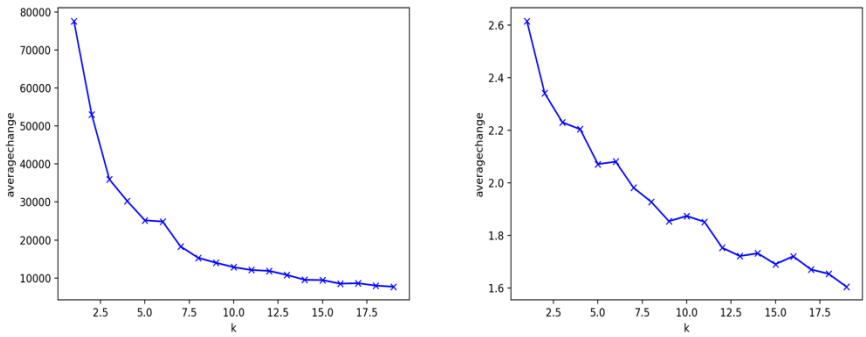
CH 评估指标
Calinski-Harabasz(CH)指标：Calinski-Harabasz 分数值 ss 越大则聚类效果越好，N=3 时,1507.902116308429

2.3MiniBirchKmeans

因为 birch 算法不适合于高维特征的聚类，因此采用 minibirchkmeans 的算法进行聚类。

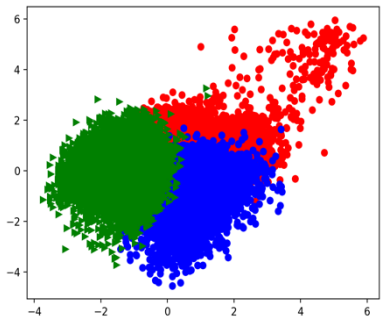
```
# mimi birch kmeans
from sklearn.cluster import MiniBatchKMeans
mbk=MiniBatchKMeans()
model=mbk.fit(loan)
labels = model.labels_
print("labels: ", labels)
cluster_centers = model.cluster_centers_
n_clusters_ = len(np.unique(labels))
print(n_clusters_)
# # #详细输出原始数据及其类别
r = pd.concat([loan, pd.Series(model.labels_, index = loan.index)], axis = 1) #详细
# # # 输出每个样本对应的类别
r.columns = list(loan.columns) + [u'聚类类别'] #重命名表头
print(r)
```

minibirchkmeans 不用提前设置聚类个数，需要通过运算比较得出 n_clusters_的个数为，这里我们依然采用肘部法则来确定聚类个数。
为了方便观察肘部曲线的值，画出了数据未标准化之前的肘部曲线，以及数据标准化后的肘部曲线
左侧图为：数据未处理前的肘部曲线，右侧图为：数据规范化后的肘部曲线



从图中可以看出 K 值从 1 到 3 时，平均畸变程度变化最大。超过 3 以后，平均畸变程度变化显著降低。因此肘部就是 K=3。

因此选取 n_clusters=3 类，降维后画出可视化结果：



Calinski-Harabasz(CH)指标 Calinski-Harabasz 分数值 ss 越大则聚类效果越好
N=3 时，4454.428816037994

3. 模型评估与分析

Calinski-Harabasz(CH)指标通过类内离差矩阵描述紧密度，类间离差矩阵描述分离度

```
#  CH 越大代表着类自身越紧密，类与类之间越分散，即更优的聚类结果
y_pred = model.fit_predict(data_zs)
print(metrics.calinski_harabaz_score(data_zs, y_pred))
```

算法	Calinski-Harabasz(CH)
Kmeans	N=3，6014
Meanshift	N=4，1507
MiniBirchKmeans	N=3，4454

根据 Calinski-Harabasz(CH)指标，可以看出 Kmeans 在分为三类时聚类效果最好，Meanshift 的聚类效果较差，而 MiniBirchKmeans 的聚类效果由于每次都是抽样进行计算，因此每次计算所得的 CH 指数均不一样，但是大致的范围均小于 5000，跟 Kmenas 相比，还是较差。

二、 实验小结

本实验依然采用第一组分类数据（美国居民收入统计数据集），希望通过分析居民的各个特征将居民进行分类，原数据集特征一共有 14 个，我们根据随机森林计算出各个特征的重要程度，进行特征筛选，删除掉影响可以忽略的特征，最终留下这八个特征：fnlwg、age、capital-gain、relationship、education-num、hours-per-week、marital-status、occupation，通过对这八个特征进行聚类分析，采用三种聚类方法：Kmenas、Meanshift、MiniBirchKmeans 来进行运算，最终通过可视化图像观察以及 CH（Calinski-Harabasz）指标来衡量不同聚类方法的效果，得出，Kmenas 在聚类数目为三是能达到较好的聚类效果，其次是 MiniBirchKmeans，但是由于 MiniBirchKmeans 每次都是进行抽样计算，因此每次计算的结果可能出现差异，但是在多次计算的比较中，可以发现该值均是大于 Meanshift 但同时又是低于 Kmeans 的，因此效果一般，最后 Meanshift 的聚类效果较差。