银行客户流失KMeans&DBSCAN比较

- 结论：KMeans的性能明显更好，可能是因为虽然KMeans和DBSCAN都是聚类算法，但客户流失问题本质上是有标签的监督学习问题，KMeans可以显示制定cluster数，而DBSCAN只能自己学习；此外，数据集质量高，noise少可能也是DBSCAN性能不理想的原因之一

数据预处理部分

1. 只使用部分相关特征

```
features = ['Age', 'CreditScore', 'EB', 'EstimatedSalary', 'Exited',
       'Gender', 'Geography', 'HasCrCard', 'IsActiveMember',
'NumOfProducts',
       'Tenure']
df = df[features]
df_test = df_test[features]
```

2. 标准化

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

KMeans

- 由于用户本质上还是“流失”/“未流失两类”，因此cluster数必定为2，不需要对k值进行调参
- 在训练集上训练模型

```
kmeans = KMeans(n_clusters=2)
kmeans.fit(X_train_scaled)
```

- 在测试集上评价聚类结果
  i. 预测类别

```
df_test['Cluster'] = kmeans.fit_predict(X_test_scaled)
```

  ii. 轮廓系数：得到0.8860

```
silhouette_avg = silhouette_score(X_test_scaled, df_test['Cluster'])
```

ii. ACC: 得到0.74

```
accuracy_score(y_test, df_test['Cluster'])
```

DBSCAN

- 与KMeans不同，DBSCAN本身没有predict方法，为了对比两个算法的性能，分别对train和test数据集进行聚类分析
- 对于训练集
  i. 建模

```
dbscan = DBSCAN(eps=0.5, min_samples=5)  # 调整eps和min_samples参数
labels = dbscan.fit_predict(X_train_scaled)
```

  ii. 簇数：44个
  分类成的簇数明显过多
  iii. 轮廓系数：-0.4372

```
silhouette_avg_test = silhouette_score(X_train_scaled, labels_test)
```

- 对于测试集
  只得到了一个簇，说明性能非常不理想