作业答案

1.使用思维导图总结聚类算法部分的内容

略

2.动手实现课程中的代码

- API案例

```
import matplotlib.pyplot as plt
import silhouette as silhouette
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
from sklearn.metrics import calinski_harabasz_score

# TODO 加载数据集
x, y = make_blobs(n_samples=100, n_features=2, centers=[[-1, -1], [0, 0],
[1, 1], [2, 2]],
                  cluster_std=[0.4, 0.2, 0.2, 0.2], random_state=22)

plt.figure()
plt.scatter(x[:, 0], x[:, 1])
plt.show()

# TODO 模型训练
model = KMeans(n_clusters=4)
y_prd = model.fit_predict(x)

plt.figure()
plt.scatter(x[:, 0], x[:, 1], c=y_prd)
plt.show()

# TODO 模型评估
print(calinski_harabasz_score(x, y_prd))
# print(silhouette.score(x, y_prd))
```

- metric案例

```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

# TODO 加载数据集
x, y = make_blobs(n_samples=1000, n_features=2, centers=[[-1, -1], [0, 0],
[1, 1], [2, 2]],
                  cluster_std=[0.4, 0.2, 0.2, 0.2], random_state=22)

# TODO 迭代不同的k值,获取sse
temp_list = []
for k in range(1, 100):
    model = KMeans(n_clusters=k, n_init='auto')
    model.fit(x)
```

```
    temp_list.append(model.inertia_)

# TODO 绘图
plt.figure()
plt.grid()
plt.plot(range(1, 100), temp_list)
plt.show()
```

- 误差平方和SSE案例

```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

# TODO 加载数据集
x, y = make_blobs(n_samples=1000, n_features=2, centers=[[-1, -1], [0, 0],
[1, 1], [2, 2]],
                  cluster_std=[0.4, 0.2, 0.2, 0.2], random_state=22)

# TODO 迭代不同的k值,获取sse
temp_list = []
for k in range(1, 100):
    model = KMeans(n_clusters=k, max_iter=100, random_state=0,
n_init='auto')
    model.fit(x)
    temp_list.append(model.inertia_)

# TODO 绘图
plt.figure(figsize=(18, 8), dpi=100)
plt.xticks(range(0, 100, 3), labels=range(0, 100, 3))
plt.grid()
plt.plot(range(1, 100), temp_list, 'or-')
plt.show()
```

- CH系数案例

```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
from sklearn.metrics import calinski_harabasz_score

# TODO 加载数据集
x, y = make_blobs(n_samples=1000, n_features=2, centers=[[-1, -1], [0, 0],
[1, 1], [2, 2]],
                  cluster_std=[0.4, 0.2, 0.2, 0.2], random_state=22)

# TODO 迭代不同的k值,获取sse
temp_list = []
for clu_num in range(2, 100):
    model = KMeans(n_clusters=clu_num, max_iter=100,
random_state=0,n_init='auto')
    model.fit(x)
    ret = model.predict(x)
```

```
    temp_list.append(calinski_harabasz_score(x, ret))

# TODO 绘图
plt.figure(figsize=(18, 8), dpi=100)
plt.xticks(range(0, 100, 3), labels=range(0, 100, 3))
plt.grid()
plt.title('ch')
plt.plot(range(2, 100), temp_list, 'or-')
plt.show()
```

- SC系数案例

```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score

# TODO 加载数据集
x, y = make_blobs(n_samples=1000, n_features=2, centers=[[-1, -1], [0, 0],
[1, 1], [2, 2]],
                  cluster_std=[0.4, 0.2, 0.2, 0.2], random_state=22)

# TODO 迭代不同的k值,获取sse
temp_list = []
for k in range(2, 100):
    model = KMeans(n_clusters=k, n_init='auto')
    model.fit(x)
    y_pre = model.predict(x)
    temp_list.append(silhouette_score(x, y_pre))

# TODO 绘图
plt.figure()
plt.grid()
plt.plot(range(2, 100), temp_list, 'or-')
plt.show()
```

- 顾客聚类分析案例

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

# TODO 加载数据
data = pd.read_csv('./data/customers.csv')
# print(data.head())

# TODO 特征选择
x = data.iloc[:, [3, 4]]
# print(x)

# TODO 模型训练
# K值选择
```

```python
sse_list = []
sc_list = []
for i in range(2, 20):
    model = KMeans(n_clusters=i, n_init='auto')
    model.fit(x)
    sse = model.inertia_
    sse_list.append(sse)
    y_pred = model.predict(x)
    sc_list.append(silhouette_score(x, y_pred))

# TODO 绘图
plt.figure()
plt.grid()
plt.plot(range(2, 20), sse_list, 'or-')
plt.show()

plt.figure()
plt.grid()
plt.plot(range(2, 20), sc_list, 'ob-')
plt.show()

# 实例化模型 K=5
model = KMeans(n_clusters=5)
model.fit(x)
y_pred = model.predict(x)
print(y_pred)
print(model.cluster_centers_)

plt.figure()
plt.scatter(x.values[y_pred == 0, 0], x.values[y_pred == 0, 1], c='r',
label='1')
plt.scatter(x.values[y_pred == 1, 0], x.values[y_pred == 1, 1], c='b',
label='2')
plt.scatter(x.values[y_pred == 2, 0], x.values[y_pred == 2, 1], c='y',
label='3')
plt.scatter(x.values[y_pred == 3, 0], x.values[y_pred == 3, 1], c='g',
label='4')
plt.scatter(x.values[y_pred == 4, 0], x.values[y_pred == 4, 1], c='gray',
label='5')
plt.scatter(model.cluster_centers_[:, 0], model.cluster_centers_[:, 1],
c='black', label='center')
plt.show()
```