数据探索与数据预处理(5)EDA_mnist

覃雄派

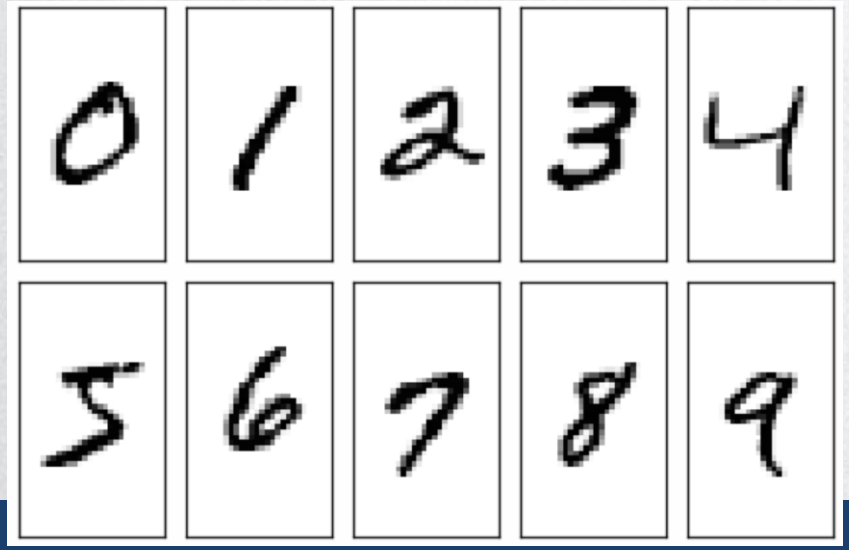
数据探索与数据预处理
(5)EDA_mnist

提纲

- MNIST数据集简介
- PCA降维简介
- T-SNE降维简介
- 降维实践

数据探索与数据预处理(5)EDA_mnist

- MNIST数据集简介
  - MNIST数据集是一个有名的手写数字数据集
  - 在机器学习领域，手写数字识别是一个很经典的学习例子
    - 原有的MNIST数据集的每个样本为28×28的点阵图片
    - 在scikit-learn库中每个样本为8×8的点阵图片



数据探索与数据预处理(5)EDA_mnist

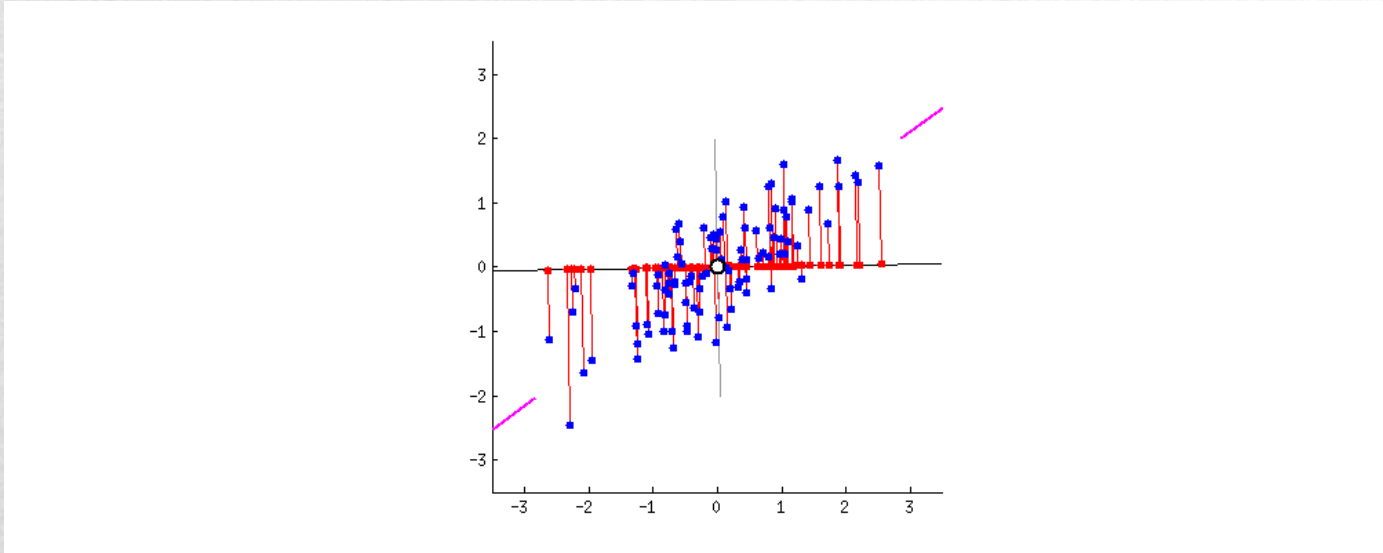
数据探索与数据预处理(5)EDA_mnist

- PCA降维简介
  - PCA的思想是将n维特征映射到k维上（k<n），这k维是全新的正交特征
  - 这k维特征称为主成分，是重新构造出来的k维特征，而不是简单地从n维特征中去除其余n-k维特征
    - 这k维的第1个维度，为原数据方差最大的方向
    - 这k维的第2个维度，为与第1维正交的情况下，方差最大的方向
    - …
    - 其它维度依此类推

PCA的数学原理和具体算法在此不展开

数据探索与数据预处理(5)EDA_mnist

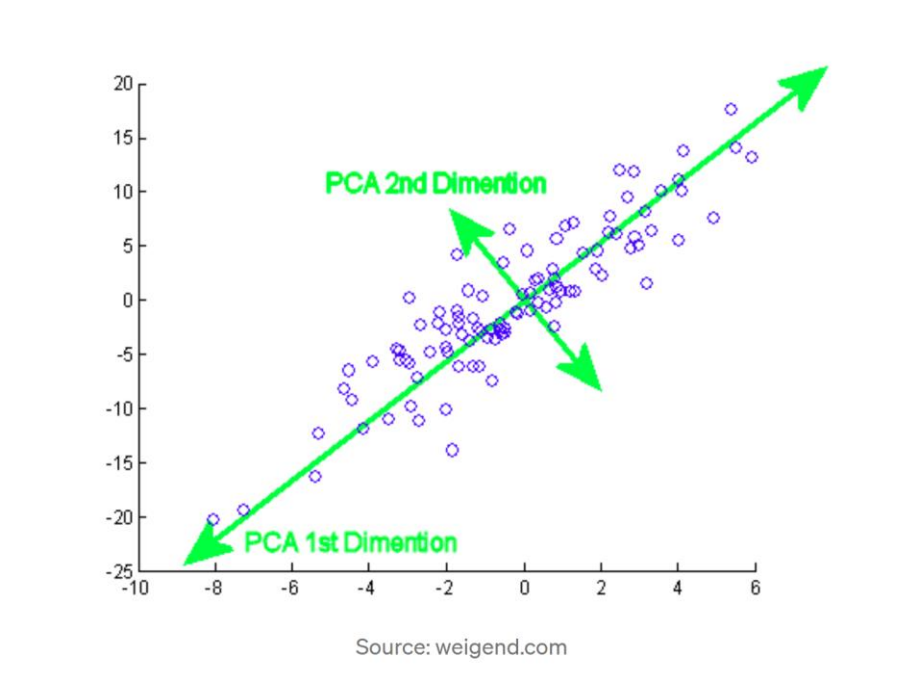
- PCA降维简介
  - 图示



这是GIF动画

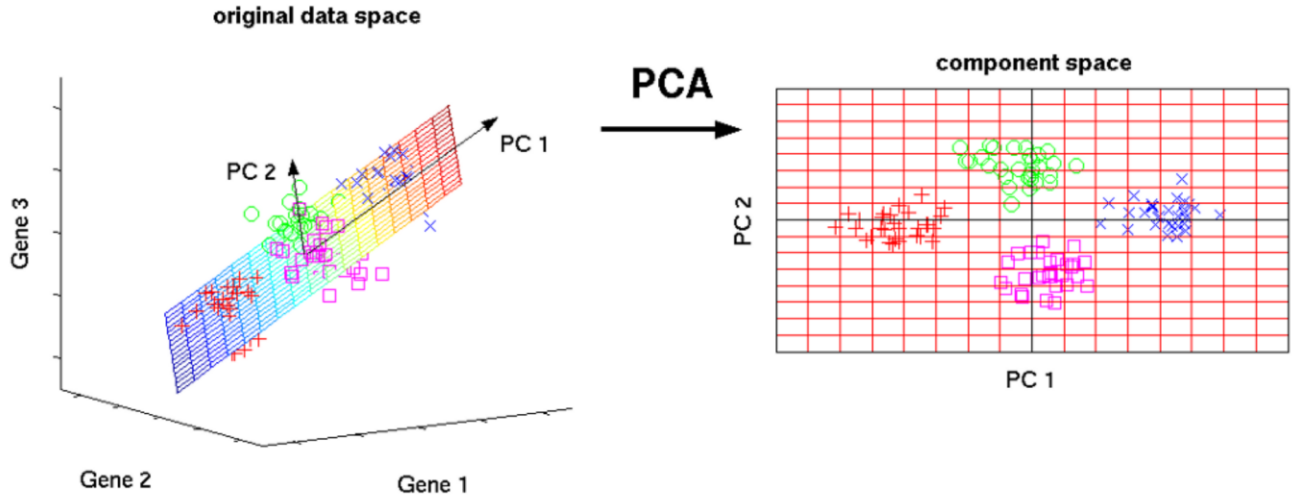
数据探索与数据预处理(5)EDA_mnist

- PCA降维简介
  - 图示



数据探索与数据预处理(5)EDA_mnist

- PCA降维简介
  - 图示



Source : A Layman's Introduction to Principal Components

数据探索与数据预处理(5)EDA_mnist

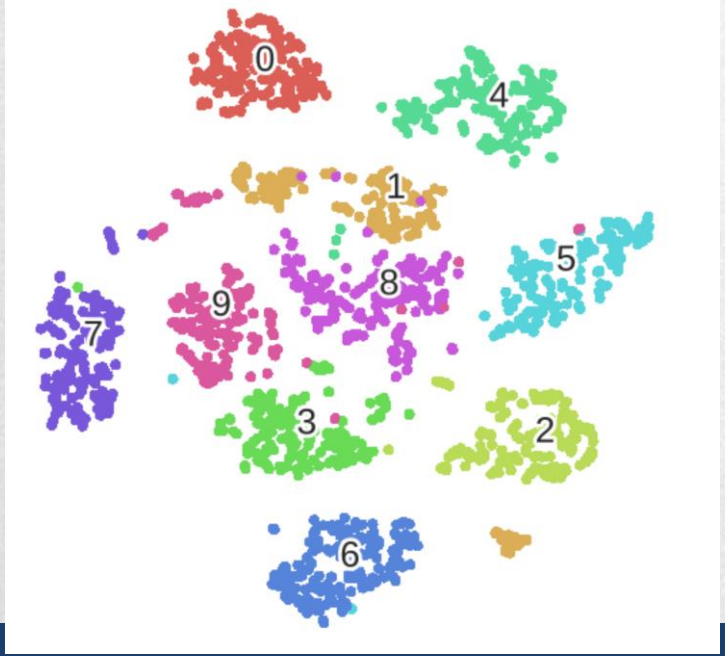
- T-SNE降维简介
  - 将高维空间分布的点的距离，用条件概率来表示相似性
  - 同时将低维空间分布的点的距离，也用条件概率来表示相似性
  - 用相对熵来训练，使得低维空间和高维空间数据点的条件概率非常接近
  - 进而把高维空间分布的点，映射到低维空间上

用通俗的话来所，经过训练，使得在高维空间距离较远的点，映射到低维空间后，距离也较远；在高维空间距离较近的点，映射到低维空间后，距离也较近

T-SNE的数学原理和具体算法在此不展开

数据探索与数据预处理(5)EDA_mnist

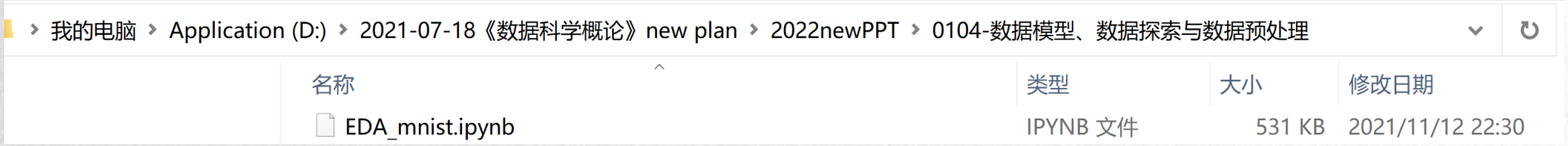
- T-SNE降维简介
  - T-SNE降维实例
  - MNIST数据集的降维效果



数据探索与数据预处理(5)EDA_mnist

数据探索与数据预处理(5)EDA_mnist

- 降维实践



数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - 装载数据集
  - 这里的每个样本是8*8的分辨率

```
# load MNIST dataset
X, y = datasets.load_digits(return_X_y=True)
print('Dimensions before PCA:', X.shape)
```

```
Dimensions before PCA: (1797, 64)
```

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - PCA方法进行降维
  - 降为2维

```
# use PCA to reduce dimension from 64 to 2
pca_2d = make_pipeline(StandardScaler(), PCA(n_components=2, random_state=0))
pca_2d.fit(X, y)
X_pca_2d = pca_2d.transform(X)
print('Dimensions after PCA-2D:', X_pca_2d.shape)
```

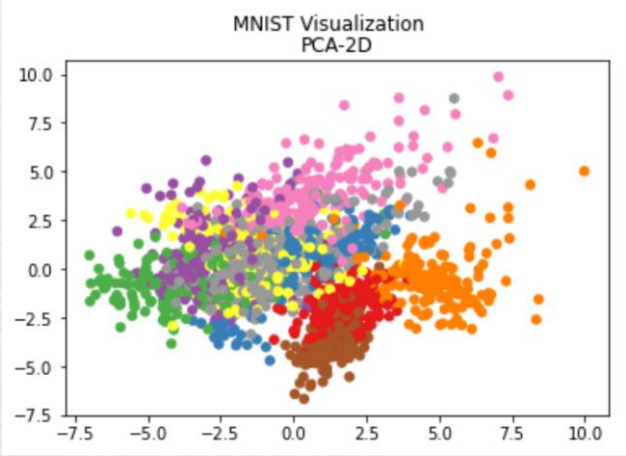
```
Dimensions after PCA-2D: (1797, 2)
```

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - PCA方法进行降维
  - 降为2维，可视化

```
# plot the points projected with PCA and tSNE
fig = plt.figure()
fig.suptitle('MNIST Visualization')

ax = fig.add_subplot(111)
ax.title.set_text('PCA-2D')
ax.scatter(X_pca_2d[:, 0], X_pca_2d[:, 1], c=y, s=30, cmap='Set1')
plt.show()
```



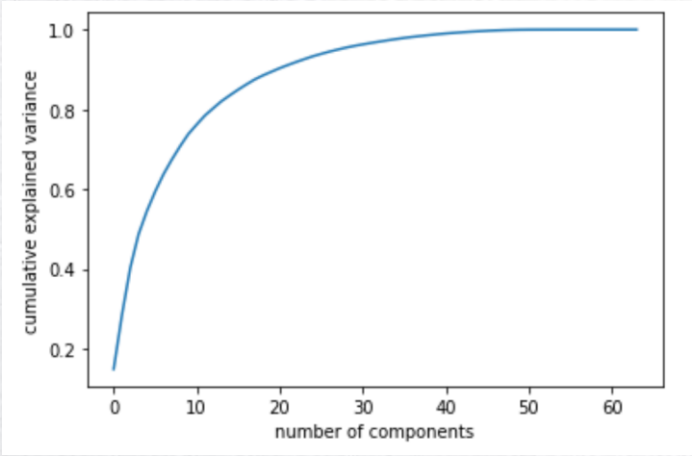
不同数字的降维向量混绞再一起

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - PCA方法进行降维
  - 查看不同components数量、和可以解释的方差的关系

```
import numpy as np

pca = PCA().fit(X)
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance');
```



数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - PCA方法进行降维
  - 降为3维

```
# use PCA to reduce dimension from 64 to 3
pca_3d = make_pipeline(StandardScaler(), PCA(n_components=3, random_state=0))
pca_3d.fit(X, y)
X_pca_3d = pca_3d.transform(X)
print('Dimensions after PCA-3D:', X_pca_3d.shape)
```

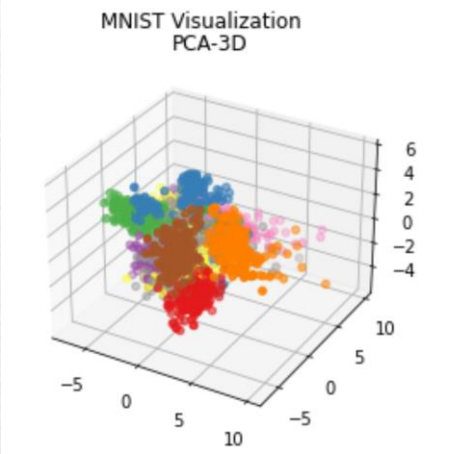
```
Dimensions after PCA-3D: (1797, 3)
```

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - PCA方法进行降维
  - 降为3维，可视化

```
fig = plt.figure()
fig.suptitle('MNIST Visualization')

ax = fig.add_subplot(111, projection='3d')
ax.title.set_text('PCA-3D')
ax.scatter(X_pca_3d[:, 0], X_pca_3d[:, 1], X_pca_3d[:, 2], c=y, cmap='Set1')
plt.show()
```



不同数字的降维向量好像分开了

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - T-SNE方法进行降维
  - 降为2维

```
# use tSNE to reduce dimension from 64 to 2
tsne = make_pipeline(StandardScaler(), TSNE(n_components=2, init='pca', random_state=0))
tsne.fit(X, y)
X_tsne_2d = tsne.fit_transform(X)
print('Dimensions after tSNE-2D:', X_tsne_2d.shape)
```

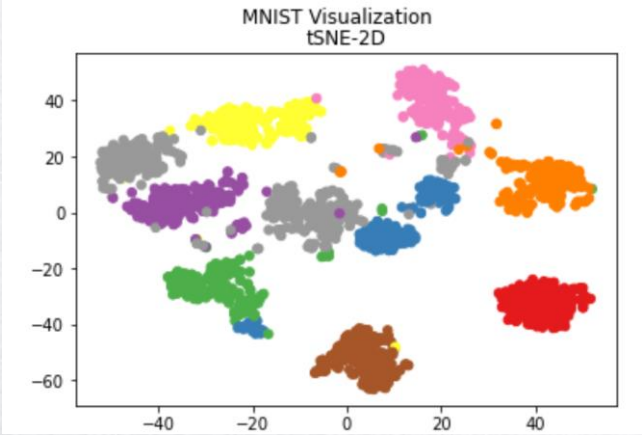
```
Dimensions after tSNE-2D: (1797, 2)
```

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - T-SNE方法进行降维
  - 降为2维，可视化

```
fig = plt.figure()
fig.suptitle('MNIST Visualization')

ax = fig.add_subplot(111)
ax.title.set_text('tSNE-2D')
ax.scatter(X_tsne_2d[:, 0], X_tsne_2d[:, 1], c=y, s=30, cmap='Set1')
```



不同数字的降维向量分开了

数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - T-SNE方法进行降维
  - 降为3维

```
# use tSNE to reduce dimension from 64 to 3
tsne = make_pipeline(StandardScaler(), TSNE(n_components=3, init='pca', random_state=0))
tsne.fit(X, y)
X_tsne_3d = tsne.fit_transform(X)
print('Dimensions after tSNE-3D:', X_tsne_3d.shape)
```

```
Dimensions after tSNE-3D: (1797, 3)
```

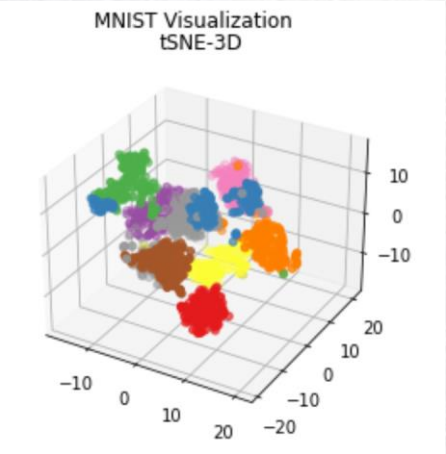
数据探索与数据预处理(5)EDA_mnist

- 降维实践
  - T-SNE方法进行降维
  - 降为3维，可视化

```
fig = plt.figure()
fig.suptitle('MNIST Visualization')

ax = fig.add_subplot(111, projection='3d')
ax.title.set_text('tSNE-3D')
ax.scatter(X_tsne_3d[:, 0], X_tsne_3d[:, 1], X_tsne_3d[:, 2], c=y, cmap='Set1')

plt.show()
```



不同数字的降维向量分开了

数据探索与数据预处理(5)EDA_mnist

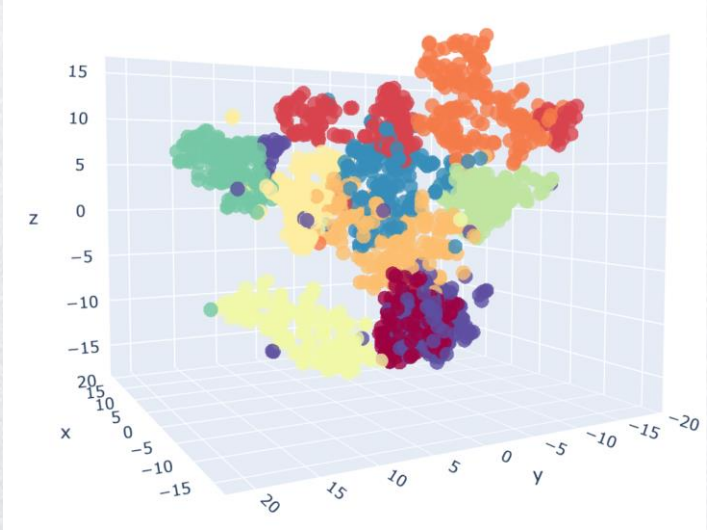
- 降维实践
  - T-SNE方法进行降维
  - （利用plotly库）降为3维，可视化

```
import plotly.graph_objects as go
```

```
xx=X_tsne_3d[:, 0]
yy=X_tsne_3d[:, 1]
zz=X_tsne_3d[:, 2]

fig = go.Figure(data=[go.Scatter3d(
    x=xx,
    y=yy,
    z=zz,
    mode='markers',
    marker=dict(
        size=6,
        color=y,                # set color to an array/
        colorscale='Spectral',   # choose a colorscale
        opacity=0.8
    )
)])

# tight layout
fig.update_layout(margin=dict(l=0, r=0, b=0, t=0))
fig.show()
```



在notebook里
可以对这个图进行旋转