Creado por:

Isabel Maniega

Análisis de Componentes Principales (PCA)

El análisis de componentes principales (PCA) es una técnica de reducción de dimensionalidad lineal que se puede utilizar para extraer información de un espacio de alta dimensión proyectándola en un subespacio de menor dimensión. Intenta preservar las partes esenciales que tienen más variación de los datos y eliminar las partes no esenciales con menos variación.

Una cosa importante a tener en cuenta sobre PCA es que es una técnica de reducción de dimensionalidad no supervisada , puede agrupar los puntos de datos similares en función de la correlación de características entre ellos sin supervisión (o etiquetas).

es un procedimiento estadístico que utiliza una transformación ortogonal para convertir un conjunto de observaciones de variables posiblemente correlacionadas (entidades cada una de las cuales toma varios valores numéricos) en un conjunto de valores de variables linealmente no correlacionadas llamadas componentes principales.

Pero, ¿dónde se puede aplicar PCA?

Visualización de datos: cuando se trabaja en cualquier problema relacionado con datos, el desafío en el mundo actual es el gran volumen de datos y las variables/características que definen esos datos. Para resolver un problema donde los datos son la clave, necesita una amplia exploración de datos, como descubrir cómo se correlacionan las variables o comprender la distribución de algunas variables. Teniendo en cuenta que hay una gran cantidad de variables o características a lo largo de las cuales se distribuyen los datos, la visualización puede ser un desafío y casi imposible.

Por lo tanto, PCA permite visualizar los datos en un espacio 2D o 3D a simple vista.

Aceleración del algoritmo de aprendizaje automático (ML): dado que la idea principal de PCA es la reducción de la dimensionalidad, puede aprovecharla para acelerar el entrenamiento y el tiempo de prueba de su algoritmo de aprendizaje automático, teniendo en cuenta que sus datos tienen muchas características y que el aprendizaje del algoritmo ML es demasiado lento.

En un nivel abstracto, toma un conjunto de datos que tiene muchas características y simplifica ese conjunto de datos seleccionando algunas Principal Componentes de las características originales.

¿Qué es un componente principal?

Los componentes principales son la clave de PCA. En términos sencillos, cuando los datos se proyectan en una dimensión más baja (suponga tres dimensiones) desde un espacio más alto, las tres dimensiones no son más que los tres componentes principales que capturan (o contienen) la mayor parte de la variación (información) de sus datos .

Los componentes principales tienen dirección y magnitud. La dirección representa a través de qué ejes principales se distribuyen principalmente los datos o tienen la mayor variación y la magnitud indica la cantidad de variación que el Componente principal captura de los datos cuando se proyecta en ese eje. Los componentes principales son una línea recta y el primer componente principal tiene la mayor variación en los datos. Cada componente principal posterior es ortogonal al último y tiene una varianza menor. De esta forma, dado un conjunto de "x" variables correlacionadas sobre "y" muestras, se obtiene un conjunto de "u" componentes principales no correlacionados sobre las mismas "y" muestras.

La razón por la que obtiene componentes principales no correlacionados de las características originales es que las características correlacionadas contribuyen al mismo componente principal, reduciendo así las características de datos originales a componentes principales no correlacionados que no representa un conjunto diferente de características correlacionadas con diferentes cantidades de variación.

Cada componente principal representa un porcentaje de la variación total capturada de los datos.

In [1]:
```
from sklearn.datasets import load_breast_cancer

breast = load_breast_cancer()
```

In [2]:
```
breast_data = breast.data
```

In [3]:
```
breast_data.shape
```

Out[3]:
```
(569, 30)
```

In [4]:
```
breast_labels = breast.target
```

In [5]:
```
breast_labels.shape
```

Out[5]:
```
(569,)
```

In [6]:
```
import numpy as np
```

In [7]:
```
labels = np.reshape(breast_labels,(569,1))
```

In [8]:
```
final_breast_data = np.concatenate([breast_data,labels],axis=1)
```

In [9]:
```
final_breast_data.shape
```

Out[9]:
```
(569, 31)
```

In [10]:
```
import pandas as pd
```

In [12]:
```
breast_dataset = pd.DataFrame(final_breast_data)
breast_dataset
```

Out[12]:

	0	1	2	3	4	5	6	7	8	9	...	21	22	23	24	25	26
0	17.990	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.16220	0.66560	0.7119
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.12380	0.18660	0.2416
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.14440	0.42450	0.4504
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.20980	0.86630	0.6869
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.13740	0.20500	0.4000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623	...	26.40	166.10	2027.0	0.14100	0.21130	0.4107
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533	...	38.25	155.00	1731.0	0.11660	0.19220	0.3215
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648	...	34.12	126.70	1124.0	0.11390	0.30940	0.3403
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016	...	39.42	184.60	1821.0	0.16500	0.86810	0.9387
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884	...	30.37	59.16	268.6	0.08996	0.06444	0.0000

569 rows × 31 columns

In [13]:
```
features = breast.feature_names
```

In [14]:
```
features
```

Out[14]:
```
array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error',
       'fractal dimension error', 'worst radius', 'worst texture',
       'worst perimeter', 'worst area', 'worst smoothness',
       'worst compactness', 'worst concavity', 'worst concave points',
       'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

In [15]:
```
features_labels = np.append(features,'label')
```

In [16]:
```
breast_dataset.columns = features_labels
```

In [17]:
```
breast_dataset.head()
```

Out[17]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0

5 rows × 31 columns

In [18]:
```
breast_dataset['label'].replace(0, 'Benign',inplace=True)
breast_dataset['label'].replace(1, 'Malignant',inplace=True)
```

In [19]:
```
breast_dataset.tail()
```

Out[19]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst are
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623	...	26.40	166.10	2027.0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533	...	38.25	155.00	1731.0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648	...	34.12	126.70	1124.0
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016	...	39.42	184.60	1821.0
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884	...	30.37	59.16	268.0

5 rows × 31 columns

In [20]:
```
from sklearn.preprocessing import StandardScaler
x = breast_dataset.loc[:, features].values
x = StandardScaler().fit_transform(x) # normalizing the features
```

In [21]:
```
x.shape
```

Out[21]:
```
(569, 30)
```

In [22]:
```
np.mean(x),np.std(x)
```

Out[22]:
```
(-6.11890932376887e-16, 1.0)
```

In [23]:
```
feat_cols = ['feature'+str(i) for i in range(x.shape[1])]
```

In [24]:
```
normalised_breast = pd.DataFrame(x,columns=feat_cols)
```

In [25]:
```
normalised_breast.tail()
```

Out[25]:

	feature0	feature1	feature2	feature3	feature4	feature5	feature6	feature7	feature8	feature9	...	feature20	feature21	fe
564	2.110995	0.721473	2.060786	2.343856	1.041842	0.219060	1.947285	2.320965	-0.312589	-0.931027	...	1.901185	0.117700	1
565	1.704854	2.085134	1.615931	1.723842	0.102458	-0.017833	0.693043	1.263669	-0.217664	-1.058611	...	1.536720	2.047399	1
566	0.702284	2.045574	0.672676	0.577953	-0.840484	-0.038680	0.046568	0.105777	-0.809117	-0.895587	...	0.561361	1.374854	0
567	1.838341	2.336457	1.982524	1.735218	1.525767	3.272144	3.296944	2.658866	2.137194	1.043695	...	1.961239	2.237926	2
568	-1.808401	1.221792	-1.814389	-1.347789	-3.112085	-1.150752	-1.114873	-1.261820	-0.820070	-0.561032	...	-1.410893	0.764190	-1

5 rows × 30 columns

In [26]:
```
from sklearn.decomposition import PCA
pca_breast = PCA(n_components=2)
principalComponents_breast = pca_breast.fit_transform(x)
```

In [27]:
```
principal_breast_Df = pd.DataFrame(data = principalComponents_breast
             , columns = ['principal component 1', 'principal component 2'])
```

In [28]:
```
principal_breast_Df.tail()
```

Out[28]:

	principal component 1	principal component 2
564	6.439315	-3.576817
565	3.793382	-3.584048
566	1.256179	-1.902297
567	10.374794	1.672010
568	-5.475243	-0.670637

In [29]:
```
print('Explained variation per principal component: {}'.format(pca_breast.explained_variance_ratio_))
```

```
Explained variation per principal component: [0.44272026 0.18971182]
```

In [30]:
```
import matplotlib.pyplot as plt
```

In [31]:
```
plt.figure()
plt.figure(figsize=(10,10))
plt.xticks(fontsize=12)
plt.yticks(fontsize=14)
plt.xlabel('Principal Component - 1',fontsize=20)
plt.ylabel('Principal Component - 2',fontsize=20)
plt.title("Principal Component Analysis of Breast Cancer Dataset",fontsize=20)
targets = ['Benign', 'Malignant']
colors = ['r', 'g']
for target, color in zip(targets,colors):
    indicesToKeep = breast_dataset['label'] == target
    plt.scatter(principal_breast_Df.loc[indicesToKeep, 'principal component 1']
                , principal_breast_Df.loc[indicesToKeep, 'principal component 2'], c = color, s = 50)

plt.legend(targets,prop={'size': 15})
```

Out[31]:
```
<matplotlib.legend.Legend at 0x7f8a595d8610>
```


Creado por:

Isabel Maniega