GUIA LABORATORIO PCA
MMD2002 - ANÁLISIS PARA CIENCIA DE DATOS

El presente laboratorio está diseñado como una guía práctica que nos ayudará a entender el valor de usar PCA para la visualización de datos. En una primera parte revisaremos algunos conceptos básicos de python para la visualización de datos, y la forma de aplicadar la técnica de PCA a través de la librería *sklearn*.

Cargar datos

El primer problema que nos encontraremos cuando se trabaje con bases de datos es poder cargar la información en memoria para poder ser utilizada. Como ejemplo, supongamos que se tiene un archivo de extensión `.csv` el cual contiene la información de hábitos alimenticios de 4 posibles países.

```
import pandas as pd
url = "datos.csv"
# load dataset into Pandas DataFrame
df = pd.read_csv(url, names=['Eng','Wal','Scot','N Ire'])
```

Pais	Queso	Carne enl	Otro tipo	Pescado	Grasas y	Azúcares	Papas	Vegetales	Otros veg	Papas pr	Vegetales	Fruta fres	Cereales	Bebidas c	Bebidas s	Bebidas a	Confiterí
ENG	105	245	685	147	193	156	720	253	488	198	360	1102	1472	57	1374	375	54
WAL	103	227	803	160	235	175	874	265	570	203	365	1137	1582	73	1256	475	64
SCOT	103	242	750	122	184	147	566	171	418	220	337	957	1462	53	1572	458	62
N IRE	66	267	586	93	209	139	1033	143	355	187	334	674	1494	47	1506	135	41

Normalizar Datos

Es importante notar que la técnica de PCA es sensible al tamaño de los datos. Esto pues el análisis se hace sobre las (co)varianzas de los variables. Para minimizar este efecto, es recomendable generar algún tipo de estándar. Para trabajar con datos existe la librería *sklearn* la cual tiene implementado el `StandardScaler`, esto es un proceso que transforma los datos en una escala unitaria de media 0 y varianza 1.

```
from sklearn.preprocessing import StandardScaler
features = ['Queso','Carne enlatada','Otro tipo de carne','Pescado','Grasas y aceites','Azucares',
...'Papas','Vegetales frescos','Otros vegetales','Papas procesadas','Vegetales procesados','Fruta fresca',
'Cereales','Bebidas calientes','Bebidas suaves','Bebidas alcoholicas','Confiteria']
# Separating out the features
x = df.loc[:, features].values
# Separating out the target
y = df.loc[:,['Pais']].values
# Standardizing the features
x = StandardScaler().fit_transform(x)
```

Pais	Queso	Carne enlatada	Otro tipo c	Pescado	Grasas y acei	Azucares	Papas	Vegetales fre	Otros veget	Papas proce	Vegetales pı	Fruta fresca	Cereales	Bebidas calie	Bebidas s	Bebidas alc	Confiteria
ENG	0.65827465	-0.0174954627	-0.259546	0.644585348025	-0.632429192	0.1305506731	-0.450075607374	0.8633118348	0.377736028	-0.33626508	0.805477228	0.7374070852	-0.644322	-0.051917413	-0.435231	0.104997662	-0.138303194
WAL	0.53580495	-1.2771687772	1.1988563	1.152440470712	1.5358994671	1.5479579816	0.435696194998	1.0935283241	1.401681627	0.084066270	1.171603241	0.9292974048	1.6794629	1.6094398081	-1.404236	0.841823362	0.9681223594
SCOT	0.53580495	-0.2274410151	0.5438111	-0.33205911867	-1.0970710471	-0.540852788	-1.335847409741	-0.7098341751	-0.4963638734	1.513192866	-0.87870243	-0.0575670951	-0.855575	-0.467256718	1.1907266	0.716562993	0.7468372487
N IRE	-1.7298845	1.5221052551	-1.4831212	-1.46496670005	0.1936007731	-1.137655866	1.350226822123	-1.2470059837	-1.2830537847	-1.26099405	-1.09837803	-1.6091373942	-0.179565	-1.090265676	0.6487407	-1.663384018	-1.576656413

PCA y proyecciones bidimensionales

Como acabamos de ver, cada una de las mediciones tiene muchas columnas, donde cada una de ellas corresponde a una variable observada. Las siguiente lineas de código proyectan los datos originales en solo dos

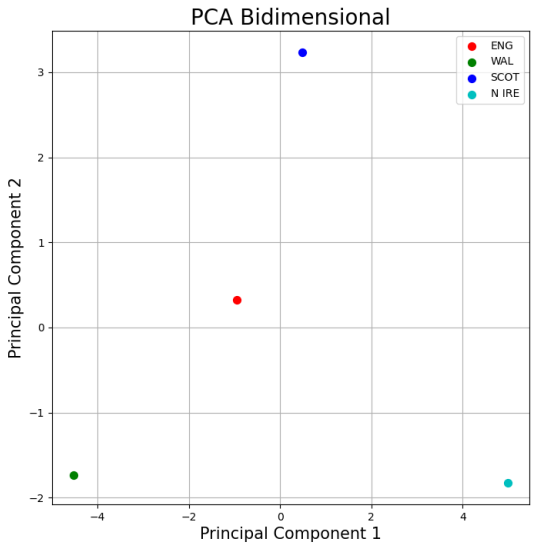
dimensiones. Es importante notar que después de esta reducción de dimensionalidad, ya no es un significado evidente en cada componente, sino que corresponden a combinaciones de las observaciones originales.

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
finalDf = pd.concat([principalDf, df[['Pais']]], axis = 1)
```

Pais	PC1	PC2
ENG	-0.954490	0.328318
WAL	-4.520951	-1.735380
SCOT	0.487978	3.233672
N IRE	4.987462	-1.826611

La ventaja de esta descripción bidimensional es que podemos fácilmente visualizar e interpretar los datos en este nuevo sistema de coordenadas:

```
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('PCA Bidimensional', fontsize = 20)
paises = ['ENG', 'WAL', 'SCOT', 'N IRE']
colors = ['r', 'g', 'b','c']
for pais, color in zip(paises,colors):
    indicesToKeep = finalDf['Pais'] == pais
    print(indicesToKeep)
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(paises)
ax.grid()
plt.savefig('ss', dpi='figure', format=None, metadata=None,
        bbox_inches=None, pad_inches=0.1,
        facecolor='auto', edgecolor='auto',
        backend=None)
plt.show()
```



Varianza explicada

La varianza explicada nos dice la cantidad de información (o varianza) que puede ser atribuida a cada componente principal. Esto es importante pues al convertir un espacio de dimensión alta, en un plano cartesiano deberíamos perder algun porcentaje de varianza del conjunto de datos. Mediante el atributo `explained_variance_ratio_` podemos ver la cantidad de varianza que cada componente principal contiene. En este caso, podemos apreciar que el 68% de la varianza está asociada a la primera componente principal, mientras que cerca del 25% corresponde a la segunda. De manera acumulada, solo el espacio bidimensional es capaz de capturar el 93% de la información contenida en la data.

Interpretación de las Componentes Principales

El último paso que podemos tratar de entender es la interpretación (cuando es posible) de las componentes principales obtenidas. Esto se puede realizar a partir del comando `components_` el cual nos da los ponderadores que construyen la combinación lineal asociada a las nuevas componentes.

```
b = pd.DataFrame(df.columns[1:])
b = pd.concat([b, pd.DataFrame(pca.components_[0,:])], axis = 1)
b = pd.concat([b, pd.DataFrame(pca.components_[1,:])], axis = 1)
b.columns = ['Variable','PC1','PC2']
```

Variable	PC1	PC2
Queso	-0.245721	0.24708
Carne enlatada	0.285629	-0.07717
Otro tipo de carne	-0.264811	0.136107
Pescado	-0.286118	-0.011006
Grases y aceites	-0.127195	-0.400545
Azucares	-0.281101	-0.136841
Papas	0.097759	-0.454688
Vegetales frescos	-0.26545	-0.096486
Otros vegetales	-0.287086	-0.092823
Papas procesadas	-0.120738	0.410361
Vegetales procesados	-0.257678	-0.15396
Fruta fresca	-0.278905	0.081746
Cereales	-0.17844	-0.329029
Bebidas calientes	-0.277475	-0.137738
Bebidas suaves	0.22772	0.29324
Bebidas alcoholicas	-0.255095	0.232318
Confiteria	-0.252758	0.211057

Ejercicio práctico

A partir de los códigos anteriores se le solicita:

(1) Visite la página link y descargue los archivos correspondientes a algunos de los años disponibles.
(2) Vaya a la pestaña llamada Anexo II. Como trabajaremos solo con una de los posibles conjuntos de datos (postulantes, seleccionados o matriculados), escoja cual de los tres le resulta de mayor interés.
(3) Construya un archivo único que incluya las distribuciones de cada una de las tablas seleccionadas, donde las filas correspondan a las Universidades y las columnas a la cantidad de individuos respecto al rango de puntaje y respecto a cada una de las pruebas.
(4) Siguiendo este instructivo, realice los pasos de cargar datos, normalizar datos, PCA y proyección bidimensional y varianza explicada.
(5) Analice cuales son las posibles diferencias entre Universidades, y explore visualmente cuales de los puntos tienen comportamientos cualitativamente diferentes entre sí.
(6) Repita los mismos experimentos anteriores con los promedios de puntajes por prueba, y en este caso simplificado haga la interpretación de las componentes principales.