```
import pandas as pd
from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler
import seaborn as sns
from sklearn.decomposition import PCA
```

```
from google.colab import drive
drive.mount('/content/gdrive/', force_remount = True)
```

```
Mounted at /content/gdrive/
```

```
df = pd.read_csv("/content/gdrive/MyDrive/stroke/country.csv", sep = ",")
```

```
df
```

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	Afghanistan	90.2	10.0	7.58	44.9	1610	9.44	56.2	5.82	553
1	Albania	16.6	28.0	6.55	48.6	9930	4.49	76.3	1.65	4090
2	Algeria	27.3	38.4	4.17	31.4	12900	16.10	76.5	2.89	4460
3	Angola	119.0	62.3	2.85	42.9	5900	22.40	60.1	6.16	3530
4	Antigua and Barbuda	10.3	45.5	6.03	58.9	19100	1.44	76.8	2.13	12200
...	...	...	...	...	...	...	...	...	...	...
162	Vanuatu	29.2	46.6	5.25	52.7	2950	2.62	63.0	3.50	2970
163	Venezuela	17.1	28.5	4.91	17.6	16500	45.90	75.4	2.47	13500
164	Vietnam	23.3	72.0	6.84	80.2	4490	12.10	73.1	1.95	1310
165	Yemen	56.3	30.0	5.18	34.4	4480	23.60	67.5	4.67	1310
166	Zambia	83.1	37.0	5.89	30.9	3280	14.00	52.0	5.40	1460

167 rows × 10 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   country     167 non-null    object
 1   child_mort  167 non-null    float64
 2   exports     167 non-null    float64
 3   health      167 non-null    float64
 4   imports     167 non-null    float64
 5   income      167 non-null    int64
 6   inflation   167 non-null    float64
 7   life_expec  167 non-null    float64
 8   total_fer   167 non-null    float64
 9   gdpp        167 non-null    int64
dtypes: float64(7), int64(2), object(1)
memory usage: 13.2+ KB
```

```
names = df[["country"]]
X = df.drop(["country"], axis = 1)
```

```
scaler = StandardScaler().fit(X)
X_scaled = scaler.transform(X)
```

```
pca = PCA(n_components=2)
pca.fit(X_scaled)
X_pca = pca.transform(X_scaled)
print(X_pca.shape)
```

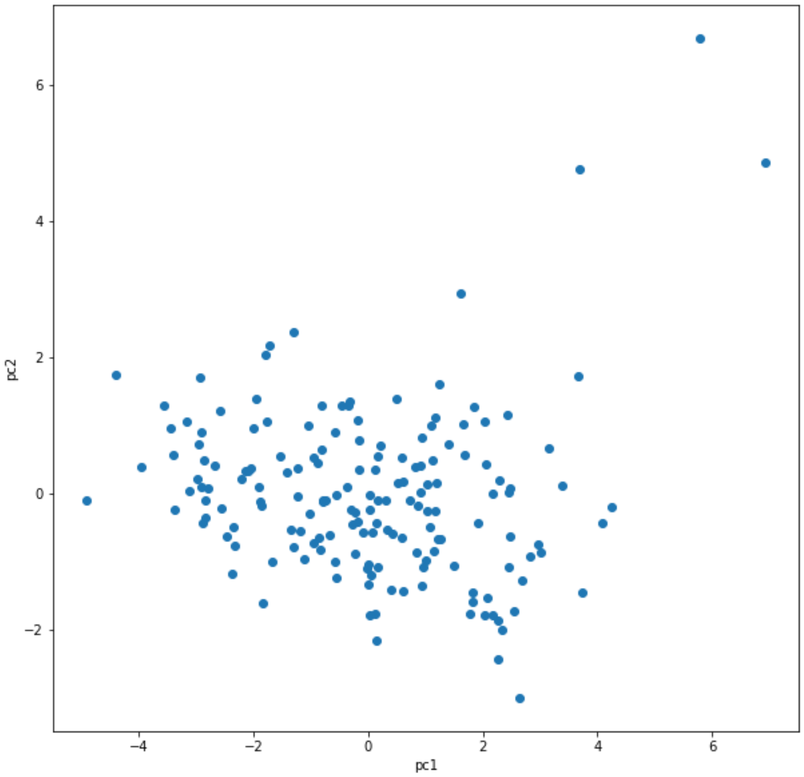
```
(167, 2)
```

```
pca.components_
```

```
array([[-0.41951945,  0.28389698,  0.15083782,  0.16148244,  0.39844111,
        -0.19317293,  0.42583938, -0.40372896,  0.39264482],
       [ 0.19288394,  0.61316349, -0.24308678,  0.67182064,  0.02253553,
        -0.00840447, -0.22270674,  0.15523311, -0.0460224 ]])
```

```
plt.figure(figsize=(10,10))
plt.scatter(X_pca[:,0],X_pca[:,1])
plt.xlabel('pc1')
plt.ylabel('pc2')
```

```
Text(0, 0.5, 'pc2')
```



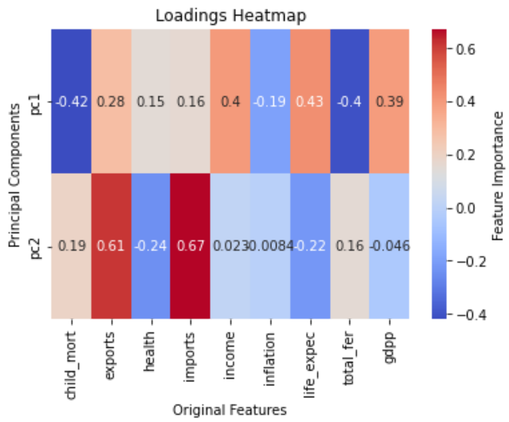

```
loadings = pca.components_
print(abs(loadings))
```

```
[[0.41951945 0.28389698 0.15083782 0.16148244 0.39844111 0.19317293
  0.42583938 0.40372896 0.39264482]
 [0.19288394 0.61316349 0.24308678 0.67182064 0.02253553 0.00840447
  0.22270674 0.15523311 0.0460224 ]]
```

```
feature_names = df.columns[1:]
feature_importance = pd.DataFrame(np.sum(loadings**2, axis=0))
feature_importance.index = feature_names
feature_importance.sort_values(0,ascending=False)
```

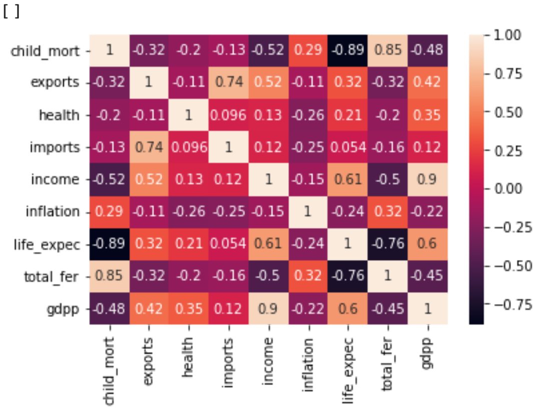
```
sns.heatmap(loadings, annot=True, cmap='coolwarm', cbar_kws={'label':'Feature Importance'}, xticklabels=feature_names, yticklabels

plt.xlabel('Original Features')
plt.ylabel('Principal Components')
plt.title('Loadings Heatmap')
plt.show()
```




```
sns.heatmap(X.corr(), annot=True)
plt.plot()
```

```
[]
```

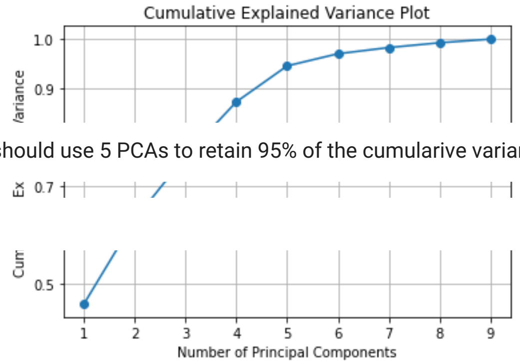


Variables that are highly correlated are assigned high weights in one of the PCAs but not the other.

```
pca = PCA(n_components=9)
X_pca = pca.fit_transform(X_scaled)
```

```
cumulative_explained_variance =  np.cumsum(pca.explained_variance_ratio_)

plt.plot(np.arange(1, len(cumulative_explained_variance) + 1), cumulative_explained_variance, marker='o')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('Cumulative Explained Variance Plot')
plt.grid()
plt.show()
```



We should use 5 PCAs to retain 95% of the cumularive variance explained.

Colab paid products - Cancel contracts here

✓ 0s completed at 11:26 PM