Question 1: Assignment Summary

Problem Statement

Categorise countries using some socio-economic and health factors that determine the overall development of the country. Then based on this categorization, list down the countries that are in direst need of aid

Solution

Use unsupervised machine learning algorithm like clustering to categorize countries and present the list of countries that need more focus

Approach

1. Check the given data and look for any missing/null values – Data was clean and there was no missing information
2. Treat the features – exports, health, imports and convert to absolute values instead of percentage
3. Look for outliers – There were many outliers. There were many countries with very high child mortality rate, high inflation, high income, low life expectancy and very high gdpp
    a. With so many outliers it would have created clusters which might not depict the correct categorisation.
    b. Since our aim is to list the countries which need more focus so we dropped the data in 1st percentile and 99th percentile
    c. With this we were able to retain 79% of data
4. Since the data present has many features like income, gdpp, child mortality, life expectancy etc. with values in different scale so its necessary to scale the data and bring the all the values at one scale
    a. For this we used Standard Scalar. This converted the data values in z-score
5. Use PCA (Principal Component Analysis) to create various principal components
    a. These components are linear combination of existing features and when combined together can explain the variance in data
    b. Use scree plots to check how many components explain 95-97% of variance in data
    c. There are total 4 components that explain 97% of variance
    d. Plot atleast 2 Principal components to visualise combination of original features in these principal components
    e. After this, the dataframe has transformed values with 4 features (principal components)
6. Use Hierarchical Clustering algorithm. With this algorithm plot the dendrogram and check how many clusters are good.
    a. Plot the cluster Ids using PCA data set and visualize the hierarchical clusters
    b. Combine these cluster Ids with original features and visualize the hierarchical clusters
    c. Use scatter plots to visualize categorization
7. Initialize K-Means algorithm with range of clusters as input and check
    a. Silhouette score, Sum of squared errors for each cluster
    b. Plot the graphs and look for optimal number of cluster
    c. k=4 was the optimal number of cluster found

8. Initialize K-Means algorithm again with input k=4 using dataframe with 4 Principal Components
    a. Plot the cluster Ids using PCA data set and visualize the k-means clusters
    b. Combine these cluster Ids with original features and visualize the k-means clusters
    c. Use scatter plots to visualize categorization
9. Compare K-Means and Hierarchical Clusters
    a. Both yielded almost similar categorization with K-Means categorization was a bit more clean with very less overlaps between clusters
    b. Choose K-Means cluster Ids as categorization for countries
10. Visualize original features data set with k-means cluster Ids and
    a. List the countries with lowest GDP per capita, highest child mortality rate and lowest income

Question 2: Clustering

a) Compare and contrast K-means Clustering and Hierarchical Clustering.

Clustering is a technique of unsupervised machine learning. There are basically two type of clustering methods:

1. K-Means Clustering algorithm
2. Hierarchical Clustering algorithm

K-Means Clustering Algorithm	Hierarchical Clustering algorithm
Its an iterative Clustering Algorithm where number of clusters is mentioned in beginning	It's also an iterative Clustering algorithm where number of clusters are not mentioned in beginning
It aims to find local maxima in each iteration	There are two different approaches 1. Agglomerative approach (Bottom to Top) 2. Divisive approach (Top to bottom)
In beginning each data point is assigned to a cluster randomly Centroid of each cluster is computed and updated Each data point is reassigned to the cluster such that the distance between each cluster centroid and data point within that cluster is minimum while the distance between centroid and each data points in other clusters is maximum The distance metric generally used is Euclidean distance	With Divisive Approach: It starts by considering all data points as one cluster and at each step divide the cluster into sub-clusters until singleton cluster of each data point is obtained With Agglomerative approach: It starts by considering each data point as individual cluster and at each step merge the closest pair of cluster The distance used is Euclidean distance Squared Euclidean distance Manhattan distance etc.
It can handle big data	It cannot handle big data
Time complexity is linear i.e. O(n)	Time complexity is quadratic i.e. $O(n^2)$

Since it always starts with random choice of initial centroids of clusters in beginning so the results produced by running this algorithm multiple times might differ	Results are always the same when this algorithm is run multiple times
Number of clusters i.e. k has to be mentioned initially	The number of cluster need not be explicitly mentioned
Silhouette Score and elbow-method can be used to determine optimal number of clusters	Dendrogram can be used to find optimal number of clusters

b) Briefly explain the steps of the K-means clustering algorithm.

K-Means Clustering Algorithm

K-Means Algorithm is an iterative clustering algorithm. It uses partitioning method to categorise data

a. For a dataset with n datapoints of m features and initial clusters given as k,
b. K-Means algorithm forms k clusters where k<=n
    a. The cost function of K-Means algorithm is given as

$$J = \sum_{i=1}^{n} ||X_i - \mu_{k(i)}||^2 = \sum_{K=1}^{K} \sum_{i \epsilon c_k} ||X_i - \mu_k||^2$$

c. In first iteration it randomly assigns each data point to a cluster
    a. The equation of assignment step is given as

$$Z_i = argmin||X_i - \mu_k||^2$$

Where X_i is the data point in the cluster and μ_k is the average of data points in that cluster

d. Then its computes the centroid of each cluster by calculating the mean of the data points assigned to each cluster. This is optimization point
    a. The equation of optimization is given as

$$\mu_k = \frac{1}{n_k} \sum_{i:z_i=k} X_i$$

Where i is the ith data point
n is the total number of data points in that cluster
k is the number of cluster

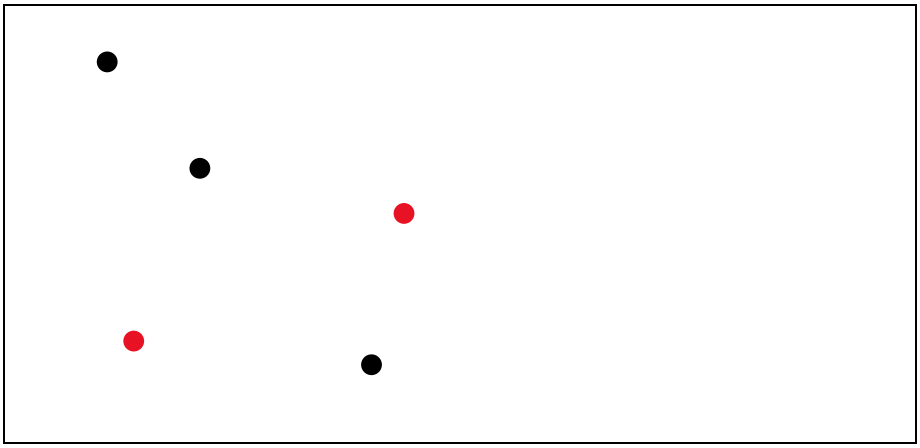
e. Then it reassigns each data point to the closest cluster such that the intra-cluster distance between each point is minimum and inter cluster distance between each point is maximum

f. Then its again computes the centroid of each cluster and updates centroid and reassigns each data point to the cluster
g. The process continues till the no more update of centroids happens

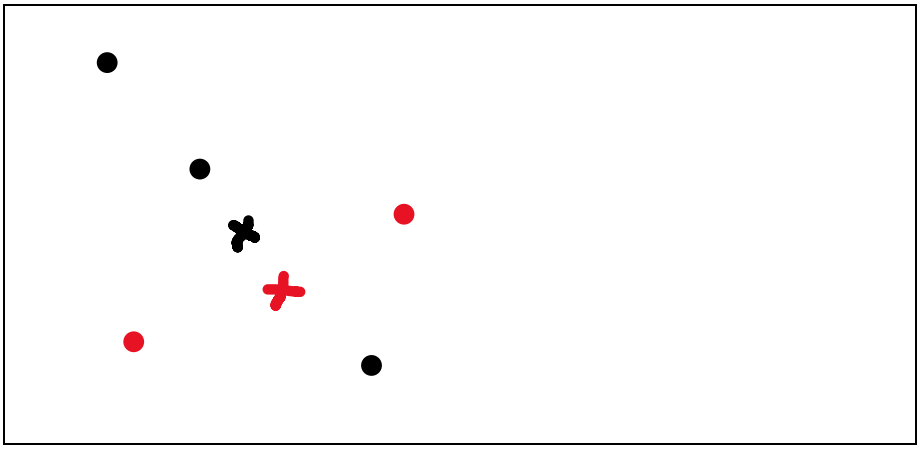
Example

K-Means algorithm with k=2

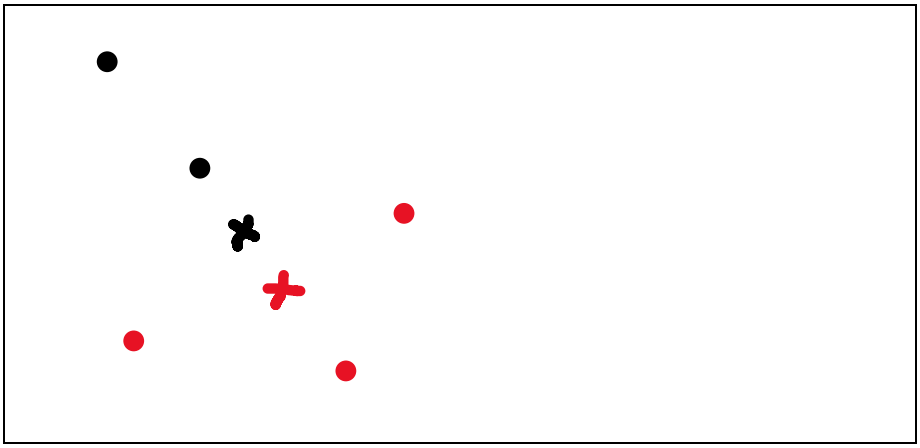
1. Initial Step



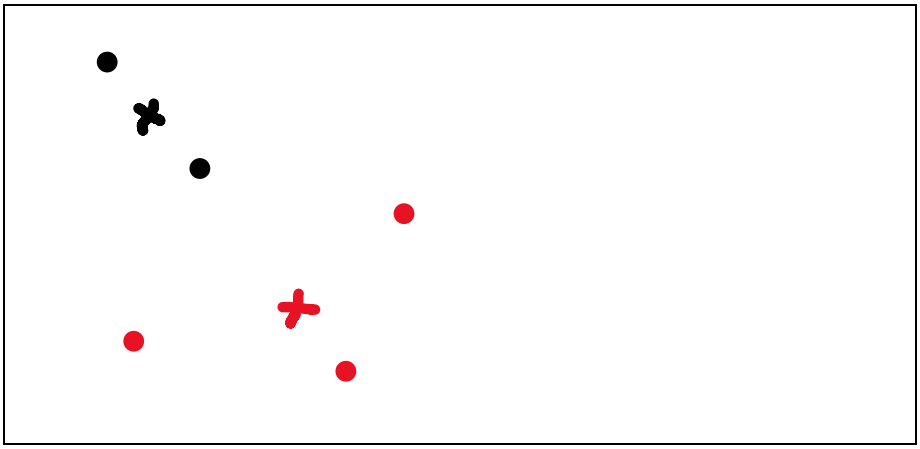
2. Compute Cluster Centroid
   Red Cross is centroid of data points in red cluster and black cross is centroid of data points in black cluster



3. Reassign data point to each closest centroid



4. Recompute cluster centroids



5. Repeat steps of 3 and 4 such that no more improvement is possible and until global optima is reached i.e. there is no further switching of data points between each cluster is possible

c) How is the value of 'k' chosen in K-means clustering? Explain both the statistical as well as the business aspect of it.

Value of k in K-means clustering can be chosen statistically by using two methods:

1. Elbow-method
2. Silhouette Score

Statistical explanation

Elbow-method

In this method, Within Clustered Sum of Squares Errors (WSS) is calculated for different number of k (clusters) and k should be chosen where WSS first starts to decrease

1. The Squared Error of each point is the square of the distance of the point from the centroid (average of all points in that cluster)

2. The WSS is the sum of squared error of all points in that cluster
3. The distance metric used can be Euclidean or Manhattan

Silhouette Score

In this method, Silhouette value measures how similar a point is to its own cluster as compared to other clusters. The range is between -1 and +1. The value closer to +1 means the point is categorised in correct cluster

The Silhouette value for each data point i is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_i| > 1$$

and

$$s(i) = 0, \text{ if } |C_i| = 1$$

Where

a. number of clusters are k and datapoint i in the cluster C_i
b. a(i) is the measure of similarity of the point i to its own cluster
c. b(i) is the measure of dissimilarity of the point i from points in other clusters

Above equation can also be written as

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

With this,

$$-1 \leq s(i) \leq 1$$

Silhouette score reaches its global maximum at optimal k

Business Aspect

Sometimes it's not clear from the elbow method of silhouette score, the optimal number of k. The numbers like k=4 or 5 etc. could have similar score or similar sum of square errors

Or sometimes, the optimal number of clusters given by system could be very large like 6 or 7 or even more.

In such cases, it becomes important to understand business needs and choose number of clusters as per business requirements so that clustering data gives better results and thus helps in drawing out inferences

Example

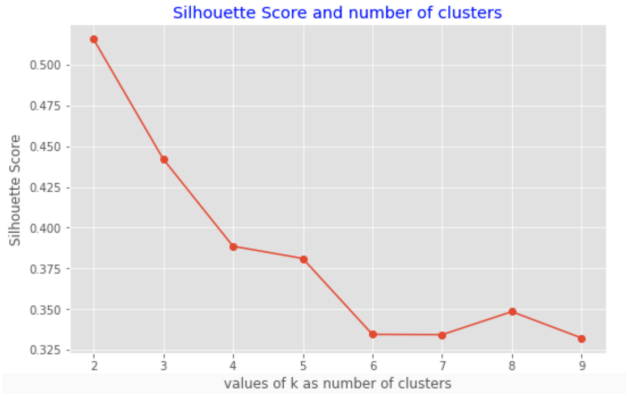
Below example shows that for k=4 is the optimal number of cluster as per the metrics however to explain to business, 4 clusters are enough to be categorised as:

i) under developed countries
ii) developing countries
iii) developed countries

So the optimal k chosen is 4 which can used for clustering, categorise the countries and draw down inference to explain in business terms

```
ssd = []
sse = []
range_n_clusters = [2,3,4,5,6,7,8,9]
for i,num_cluster in enumerate(range_n_clusters):
    kmeans = KMeans(n_clusters=num_cluster)
    kmeans.fit(country_pca_df)
    ssd.append(kmeans.inertia_)
    ##silhouette score
    silhouette_avg = silhouette_score(country_pca_df,kmeans.labels_)
    sse.append(silhouette_avg)
    print("For n_clusters={0}, the silhouette score is {1} and squared error is {2}".
          format(num_cluster, silhouette_avg, kmeans.inertia_.round(2)))

plt.figure(figsize=(8,10))
plt.subplot(2,1,1)
plt.plot(ssd,marker='o');
plt.title('Cost and number of clusters',color='blue')
plt.xlabel('values of k as number of clusters')
plt.ylabel('Squared Error (Cost)');
plt.xticks(np.arange(len(range_n_clusters)), range_n_clusters);
plt.tight_layout(pad=3.0)
plt.subplot(2,1,2)
plt.plot(sse,marker='o');
plt.title('Silhouette Score and number of clusters',color='blue')
plt.xlabel('values of k as number of clusters')
plt.ylabel('Silhouette Score');
plt.xticks(np.arange(len(range_n_clusters)), range_n_clusters);
```

```
For n_clusters=2, the silhouette score is 0.5157343170623355 and squared error is 588.58
For n_clusters=3, the silhouette score is 0.4421125006356147 and squared error is 359.35
For n_clusters=4, the silhouette score is 0.35684661694331327 and squared error is 297.99
For n_clusters=5, the silhouette score is 0.29981581747665137 and squared error is 253.07
For n_clusters=6, the silhouette score is 0.3250507505577652 and squared error is 212.61
For n_clusters=7, the silhouette score is 0.35121957635775425 and squared error is 182.68
For n_clusters=8, the silhouette score is 0.35169961185442644 and squared error is 156.17
For n_clusters=9, the silhouette score is 0.3319072787238899 and squared error is 138.81
```



d) Explain the necessity for scaling/standardisation before performing Clustering.

Scaling or Standardization is necessary before clustering because otherwise the range of values in each feature will act as a weight when determining how to cluster with features with more values will dominate and therefore might not produce correct results

Scaling or Standardization is the process of rescaling the values of the variables in data set so they share a common scale. It's a pre-processing step
For cluster analysis, standardization may be important if each feature/attribute in data set has a different unit (e.g., inches, meters, tons and kilograms), or where the scales of each feature/attribute are very different from one another (e.g., 0-1 vs 0-1000).

K-Means algorithm uses Euclidean distance as metric which is produce different results when data is not scaled. The features with large values will dominate the measure. The range of values in each feature will act as a weight and if the data is not scaled then the features with values on large scale will dominate over the features with values on small scale like inches vs kilograms

In a situation where one feature/attribute has a much greater range of value than another (because the field with the wider range of values likely has greater distances between values), it may end up being the primary driver of what defines clusters.
Standardization helps to make the relative weight of each variable equal by converting each variable to a unitless measure or relative distance.

```python
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.metrics.pairwise import euclidean_distances

X = np.array([[1,2,100],
              [4,3,50],
              [1,1,75]])

print('Matrix X before scaling: \n',X.round(2))

print('\neuclidean distance before scaling: \n',euclidean_distances(X).round(2))

sc = StandardScaler()
X_sc = sc.fit_transform(X)

print('\nMatrix X after scaling: \n',X_sc.round(2))

print('\neuclidean distance after scaling: \n',euclidean_distances(X_sc).round(2))
```

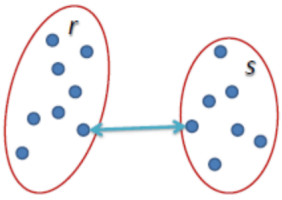
```
Matrix X before scaling:
 [[  1   2 100]
 [  4   3  50]
 [  1   1  75]]

euclidean distance before scaling:
 [[ 0.   50.1  25.02]
 [50.1   0.   25.26]
 [25.02 25.26  0.  ]]

Matrix X after scaling:
 [[-0.71  0.    1.22]
 [ 1.41  1.22 -1.22]
 [-0.71 -1.22  0.  ]]

euclidean distance after scaling:
 [[0.   3.46 1.73]
 [3.46 0.   3.46]
 [1.73 3.46 0.  ]]
```

In above example it can be seen that, column 3 before standardization is more important since it has more range of values while its not the case after standardization

e) Explain the different linkages used in Hierarchical Clustering.

There are three types of linkages in Hierarchical Clustering

1. Single Linkage
    a. In this type of linkage the distance between two clusters is defined as the shortest distance between points in two clusters
    Example

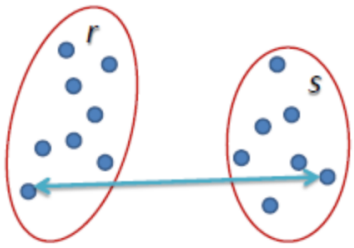


$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Where r and s are two clusters
And x_{ri} is the ith data point in cluster r and x_{sj} is the jth data point in cluster s
And D is the distance between data points

Thus the minimum distance between data points in two clusters r and s defines the single linkage here

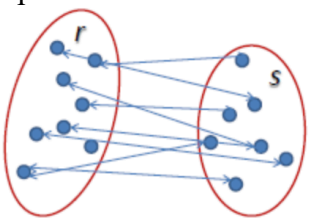
2. Complete Linkage
    a. In this type of linkage the distance between the two clusters is defined as the maximum distance between any two points in the clusters
    Example



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Where r and s are two clusters
And x_{ri} is the ith data point in cluster r and x_{sj} is the jth data point in cluster s
And D is the distance between data points
Thus the maximum distance between data points in two clusters r and s defines the complete linkage here

3. Average Linkage
    a. In this type of linkage the distance between the two clusters is defined as the average distance between every point of one cluster to every other point of another cluster
    Example



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Where r and s are two clusters
And x_{ri} is the ith data point in cluster r and x_{sj} is the jth data point in cluster s
And D is the distance between data points
And n_r is the number of data points in cluster r and n_s is the number of data points in cluster s
Thus the average distance of each point in cluster r to each data point in cluster s defines the average linkage here

Question 3: Principal Component Analysis

a) Give at least three applications of using PCA.

Principal Component Analysis is the dimensionality reduction technique and when applied on data set with dimensions M*n results in dimensions M*k where k is the number of principal components that explain most of the variance in data. Principal components are the linear combination of original features

Principal components have following properties

1. Dimensionality Reduction:
    a. For high dimensional data, PCA allows to reduce the dimensionality of data so that the majority of the variation in data across many high dimensions is captured in fewer dimensions
2. Orthogonal:
    a. The Principal Components are orthogonal i.e. uncorrelated to each other. In other words, the correlation between any pair of principal component is 0
3. Eigenvectors:
    a. The eigenvectors are represented by principal components
    b. Consider a non-zero vector v. Its an eigenvector of a square matrix A, if Av is a scalar multiple. In other words
        i. Av = λv
        ii. Where v is the eigenvector and λ is the eigenvalue associated with it
4. The variation present in the principal components decreases as we move from first principal component to the last principal component

The three applications of using PCA are

1. Cluster Analysis
2. Plotting relationships
3. Recommendations engines (which uses cosine similarity as the metrics)

b) Briefly discuss the 2 important building blocks of PCA - Basis transformation and variance as information.

Basis Transformation is a technique applied in a vector space in order to re-write the vector in the form of different set of basis elements

Suppose

V is the linear transformation of the matrix N in basis B1 = {v1,v2…..,vn}
and V has another basis B2 = {u1,u2…..,un}
If M is the change of basis matrix for B1 to B2, then the linear transformation T is the output expressed in basis B2 and can be written as

$$T(u) = MNM^{-1}$$

PCA can be considered as a process of finding a new basis which is a linear combination of the original basis

Example

Components in below example represent the eigen vectors or in other terms as principal components of given data set with original feature set as child_mort, exports, health, imports, income, inflation, life_expec, total_fer, gdpp

```
country_pca = pd.DataFrame({'Feature':numerical_cols,
                            'PC1':pca.components_[0],
                            'PC2':pca.components_[1],
                            'PC3':pca.components_[2],
                            'PC4':pca.components_[3]})
country_pca
```

	Feature	PC1	PC2	PC3	PC4
0	child_mort	-0.419519	0.192884	-0.029544	0.370653
1	exports	0.283897	0.613163	0.144761	0.003091
2	health	0.150838	-0.243087	-0.596632	0.461897
3	imports	0.161482	0.671821	-0.299927	-0.071907
4	income	0.398441	0.022536	0.301548	0.392159
5	inflation	-0.193173	-0.008404	0.642520	0.150442
6	life_expec	0.425839	-0.222707	0.113919	-0.203797
7	total_fer	-0.403729	0.155233	0.019549	0.378304
8	gdpp	0.392645	-0.046022	0.122977	0.531995

Variance as information

Total variance is the sum of variances of all individual principal components

The fraction of variance explained by a principal component is the ratio between the variance of that principal component and total variance

PCA computes the new set of variables and the data is represented in terms of these new components. When combined together, the new principal components represent the same amount of information as the original variables/features. The total variance remains the same. However, variances is distributed amongst the new components such that first component always explain the most variance amongst the other components and this variance keeps on decreasing from 1st till last component. Generally first k principal components (where k can be 1,2,3 etc) explain the most variance

Scree plot can be used to choose the optimal number of components that explain maximum variance when combined together

Example

```
In [229]:
pca = PCA(svd_solver='randomized',random_state=42)
country_pca_df = pca.fit(country_sc_df[numerical_cols].values)
```

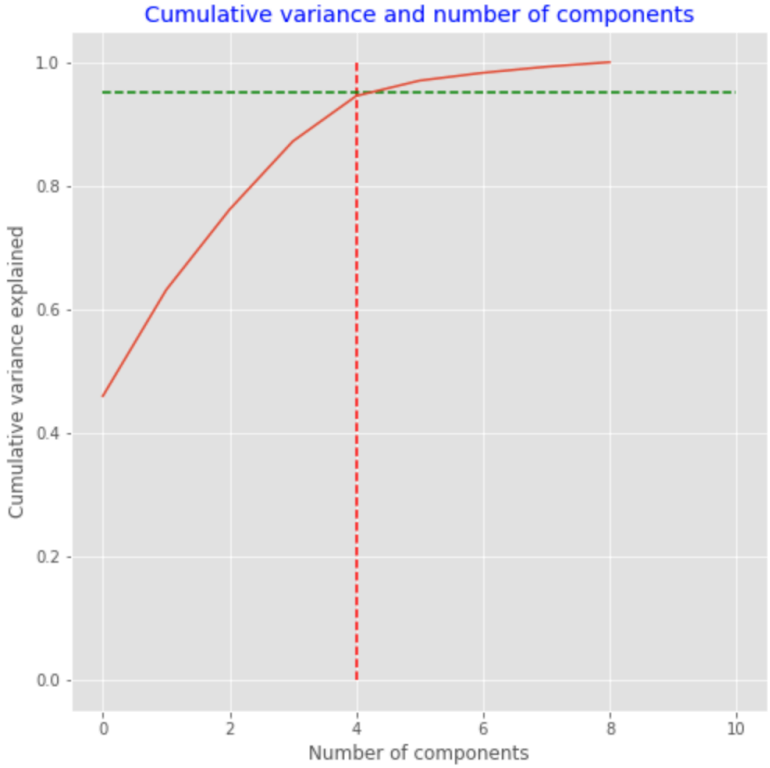
```
In [230]:
pca.explained_variance_ratio_
```

```
Out[230]: array([0.4595174 , 0.17181626, 0.13004259, 0.11053162, 0.07340211,
                 0.02484235, 0.0126043 , 0.00981282, 0.00743056])
```

Scree-Plot Example

Below example shows that 95% of the variance can be explained by 4 components

```
plt.figure(figsize=(8,8));
plt.plot(np.cumsum(pca.explained_variance_ratio_));
plt.ylabel('Cumulative variance explained')
plt.xlabel('Number of components')
plt.title('Cumulative variance and number of components',color='blue')
plt.vlines(x=4, ymax=1, ymin=0, colors="r", linestyles="--")
plt.hlines(y=0.95, xmax=10, xmin=0, colors="g", linestyles="--");
```



c) State at least three shortcomings of using Principal Component Analysis.

1. Loss of interpretability:
    a. Principal Components are linear combination of original features. Due to this the independent variables become less interpretable and no inference wrt original feature can be deduced
2. Data Standardization is must for PCA
    a. The data must be scaled/standardized before applying PCA. For instance, if a feature set has data set in units of weight in kgs., height in inches, amount in millions, then the variance scale is huge in the training set. If PCA is applied on such a feature set, the resultant for features with high variance will also be large. Hence, principal components will be biased towards features with high variance, leading to false results.
3. PCA is limited to linearity. For non-linear technique t-SNE should be used
4. PCA needs components to be orthogonal which might not be true in some cases. For such cases Independent Component Analysis must be used

5. PCA assumes columns with low variance are not important which might not be true always especially in the classification problems with highly imbalanced target class data set