Name : Priyangana Das
Roll No : 301911001001
Class : B.E.I.T 4th year 1st semester

ML Lab Assignment : 4

1) *Partition based: K-means*

IRIS PLANT DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
```

```
iris=load_iris()   #loading iris dataset from sklearn.datasets
iris
```

```
df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df
```

```
x=iris.data
```

```
plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)
```

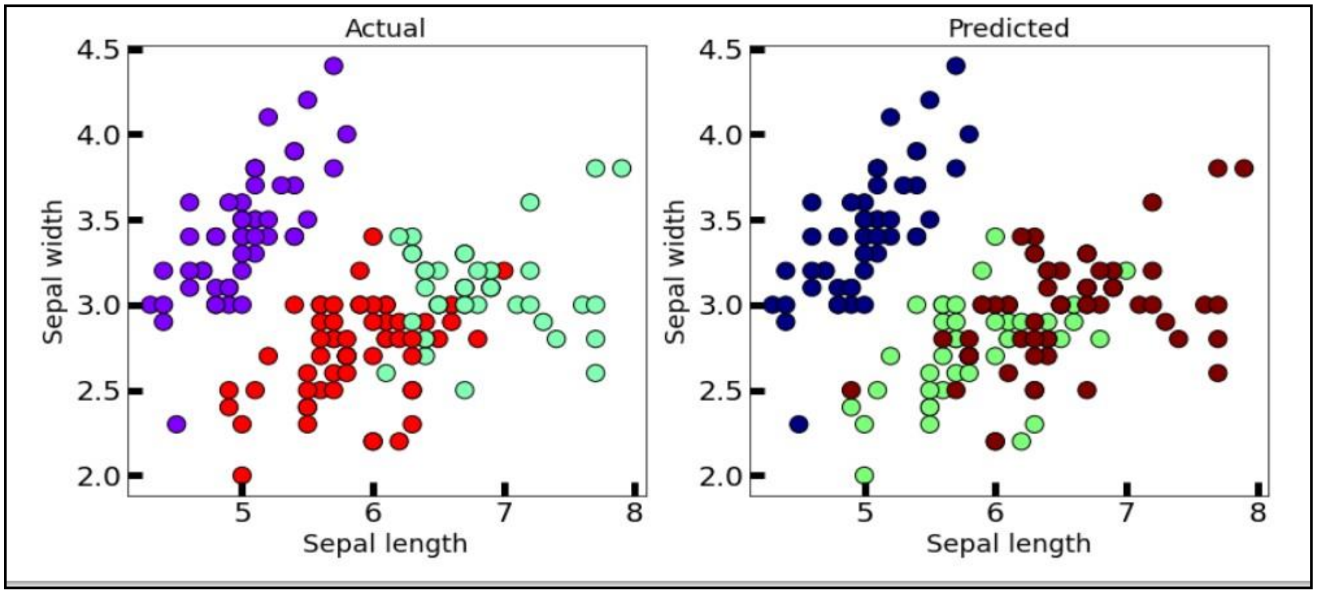
```
kmeans = KMeans(init="random", n_clusters=3, n_init=10, max_iter=300, random_state=42)
y = kmeans.fit_predict(x)
```

```
print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)
```

```
plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='
rainbow'
plt.show()
```

```
fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
```

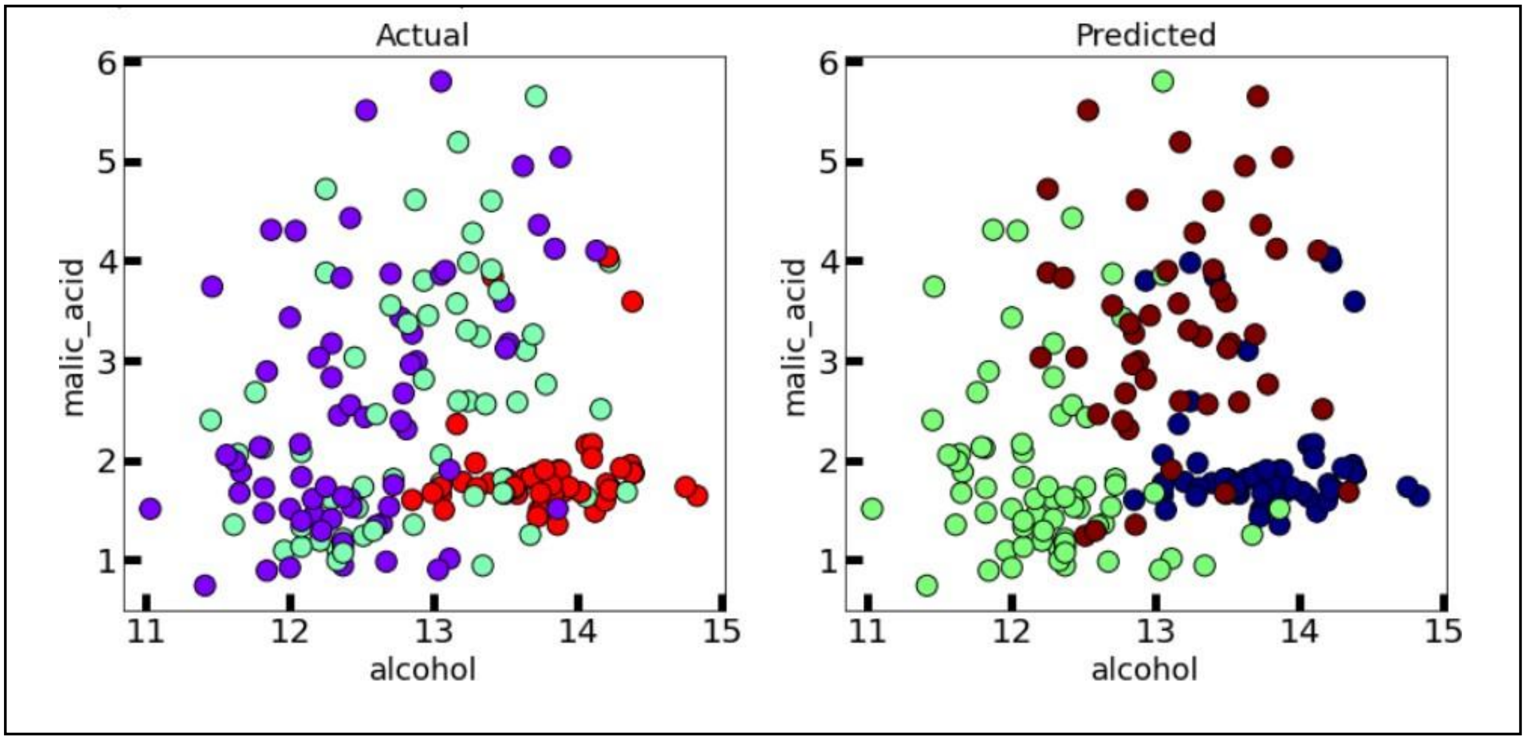
```
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)


from sklearn.metrics import silhouette_score
print("The silhouette score is :")
silhouette_score(x, kmeans.labels_)


from sklearn.metrics import calinski_harabasz_score
print("The calinski harabasz score is :")
calinski_harabasz_score(x, kmeans.labels_)


from sklearn.metrics import davies_bouldin_score
print("The davies bouldin score is :")
davies_bouldin_score(x, kmeans.labels_)


print("It is observed that TSS=SSE+SSB is a constant. Hence we will calculate the TSS ans substract SSE fr
om it to get SSB")

print("The value of SSE is: ")
print(kmeans.inertia_)



# Finding the overall centroid of the data points
centers = kmeans.cluster_centers_
center_x = []
for center in centers:
  center_x.append(center[0])
center_x
overall_center = sum(center_x)/len(center_x)

tss = 0
for i in range(len(df)):
  a = df.iloc[i][0] - overall_center
  b = pow(a,2)
  tss = tss+b



print("The value of SSB is: ")
print(tss - kmeans.inertia_)
```



WINE DATASET

```
 #importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine


wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'
```

```
plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)

kmeans = KMeans(init="random", n_clusters=3, n_init=10, max_iter=300, random_state=42)
y = kmeans.fit_predict(x)

print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='rainbo
w'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)

from sklearn.metrics import silhouette_score
print("The silhouette score is :")
silhouette_score(x, kmeans.labels_)

from sklearn.metrics import calinski_harabasz_score
print("The calinski harabasz score is :")
calinski_harabasz_score(x, kmeans.labels_)

from sklearn.metrics import davies_bouldin_score
print("The davies bouldin score is :")
davies_bouldin_score(x, kmeans.labels_)

print("It is observed that TSS=SSE+SSB is a constant. Hence we will calculate the TSS ans substract SSE fr
om it to get SSB")

print("The value of SSE is: ")
print(kmeans.inertia_)



# Finding the overall centroid of the data points
centers = kmeans.cluster_centers_
center_x = []
for center in centers:
  center_x.append(center[0])
center_x
overall_center = sum(center_x)/len(center_x)

tss = 0
for i in range(len(df)):
  a = df.iloc[i][0] - overall_center
  b = pow(a,2)
  tss = tss+b



print("The value of SSB is: ")
print(tss - kmeans.inertia_)
```



This algorithm generalizes to clusters of different shapes and sizes, such as elliptical clusters. The problem with it is that we need to manually choose the value of “k”.

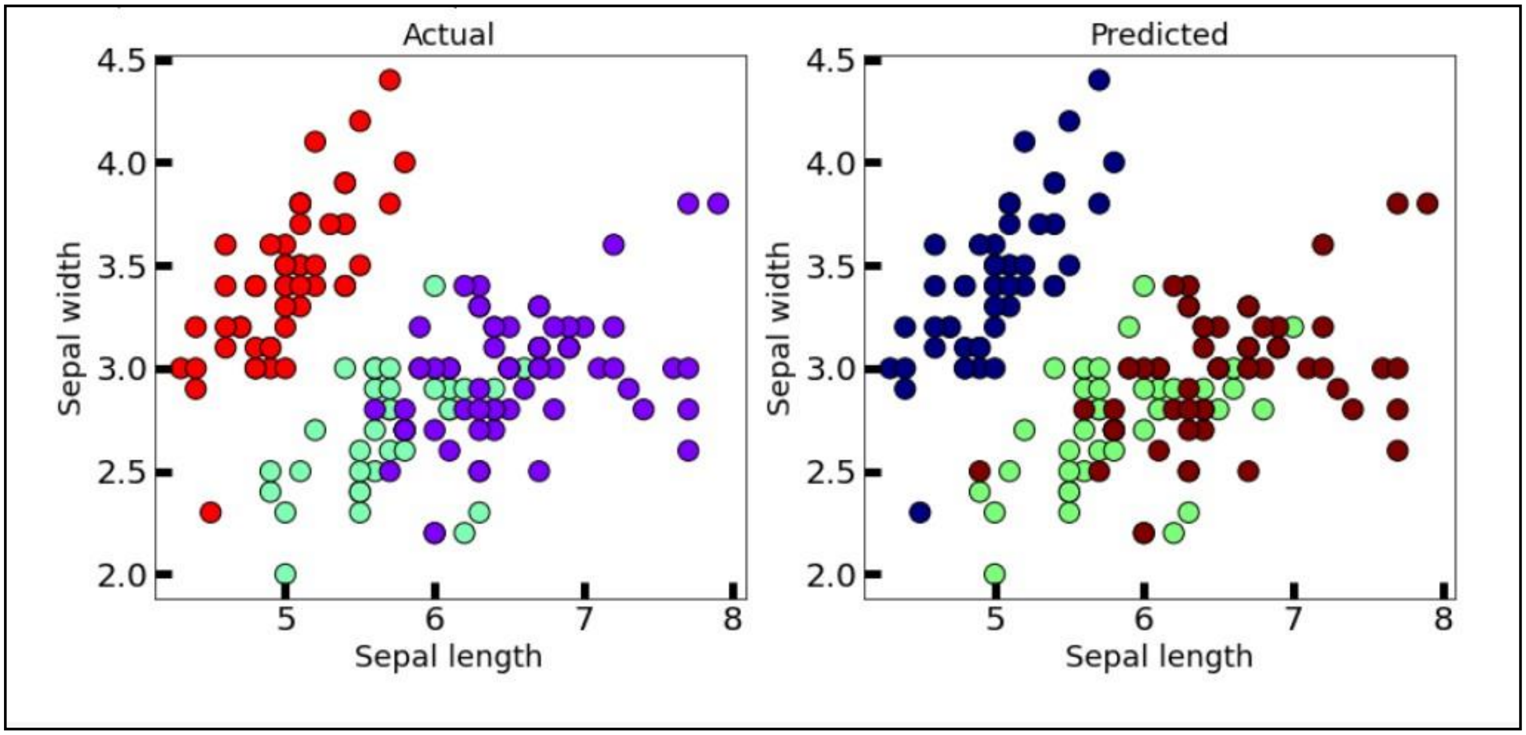
2) Partition based: K-medoids

IRIS PLANT DATASET

```
 #importing libraries
!pip install scikit-learn-extra
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn_extra.cluster import KMedoids
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris
df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])



x=iris.data



plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)



kmedoid = KMedoids(init="heuristic", n_clusters=3, max_iter=300, random_state=42)
y = kmedoid.fit_predict(x)



print("K-Medoids Cluster Centers")
print(kmedoid.cluster_centers_)
print("Cluster Labels")
print(kmedoid.labels_)



plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=kmedoid.labels_, cmap='rainbow') #try using cmap=
'rainbow'
plt.show()



fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
```

```
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```

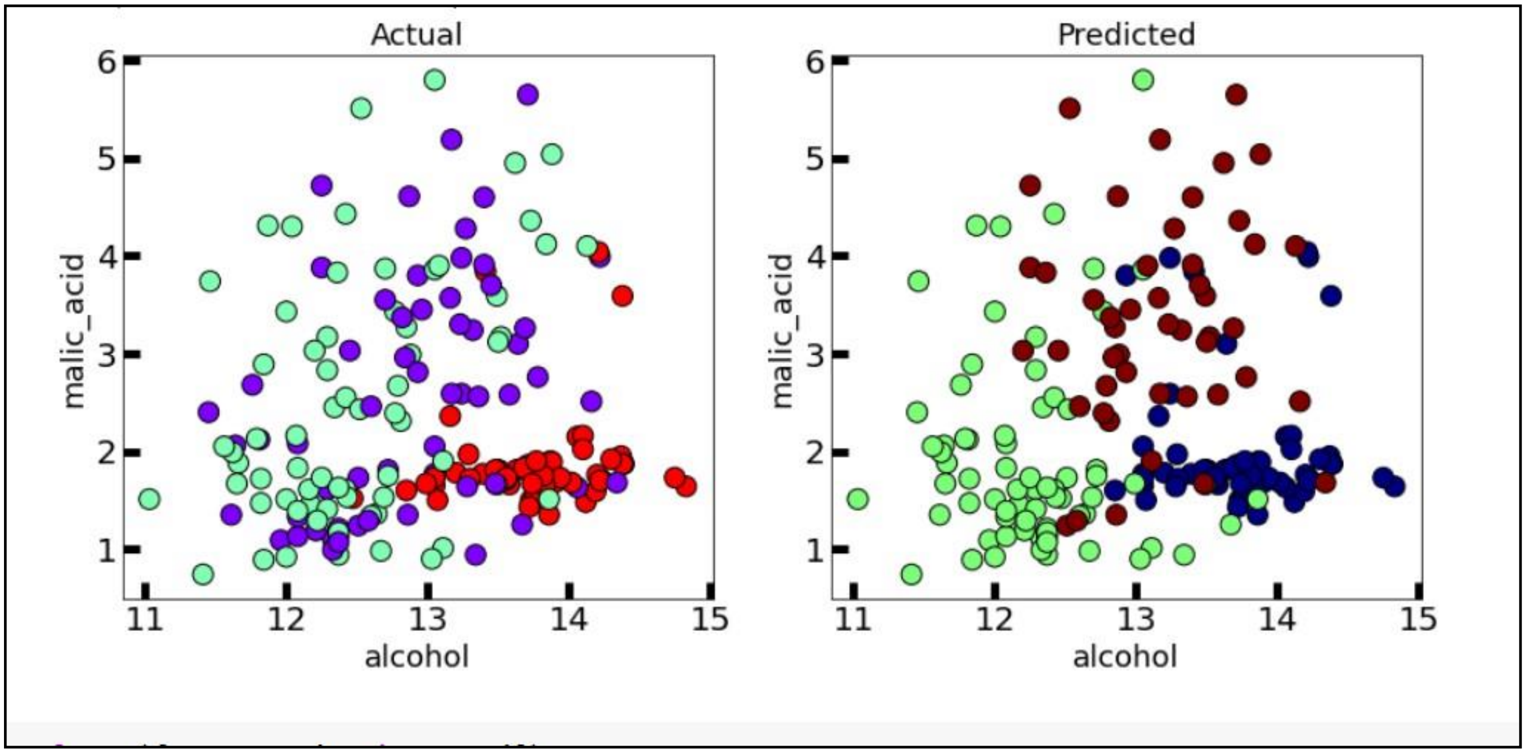
Also added some scores code .



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn_extra.cluster import KMedoids
from sklearn.datasets import load_iris
```

```
wine=load_wine()   #loading iris dataset from sklearn.datasets
wine


x=wine.data


df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df


plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)


kmedoid = KMedoids(init="heuristic", n_clusters=3, max_iter=300, random_state=42)
y = kmedoid.fit_predict(x)


print("K-Medoids Cluster Centers")
print(kmedoid.cluster_centers_)
print("Cluster Labels")
print(kmedoid.labels_)


plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=kmedoid.labels_, cmap='rainbow') #try using cmap='rainb
ow'
plt.show()


fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



This algorithm solves the problem with the K-means algorithm.

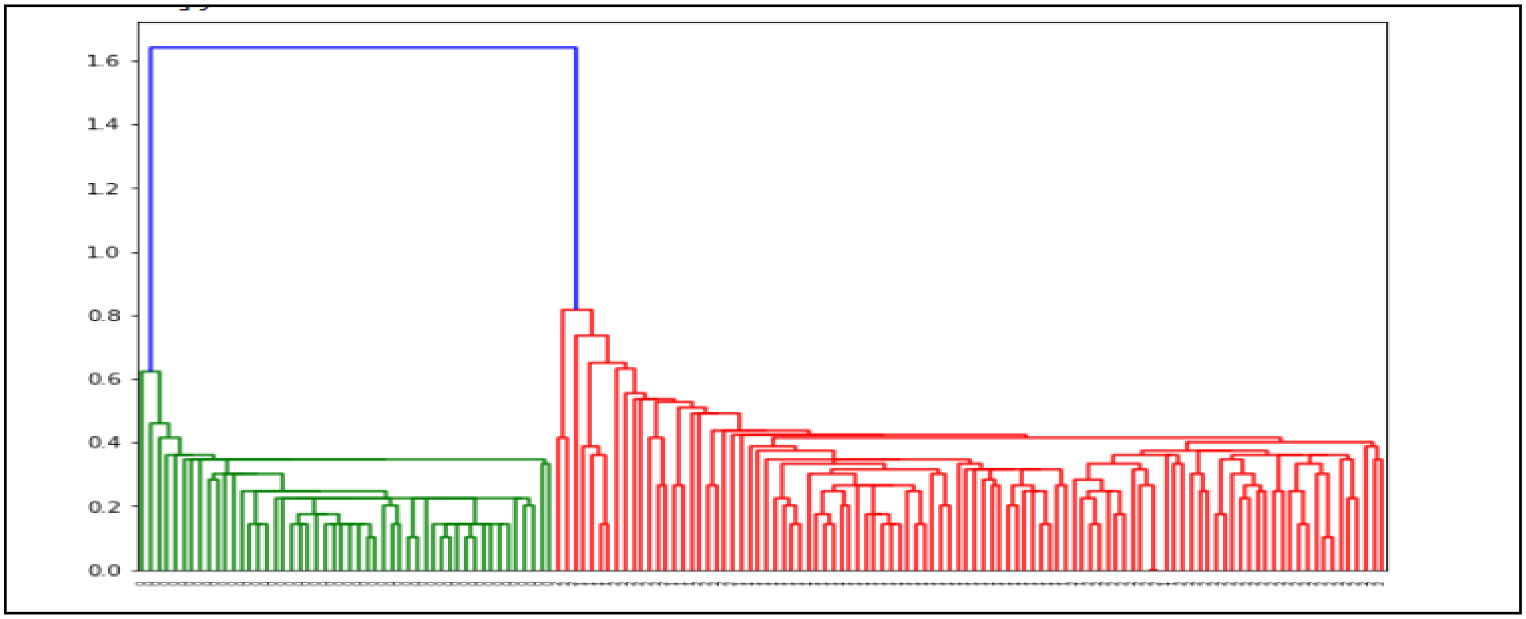
K-means attempts to minimize the total squared error, while k-medoids minimizes the sum of dissimilarities between points labeled to be in a cluster and a point designated as the center of that cluster..

3) *Hierarchical: Dendrogram*

IRIS PLANT DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data
```

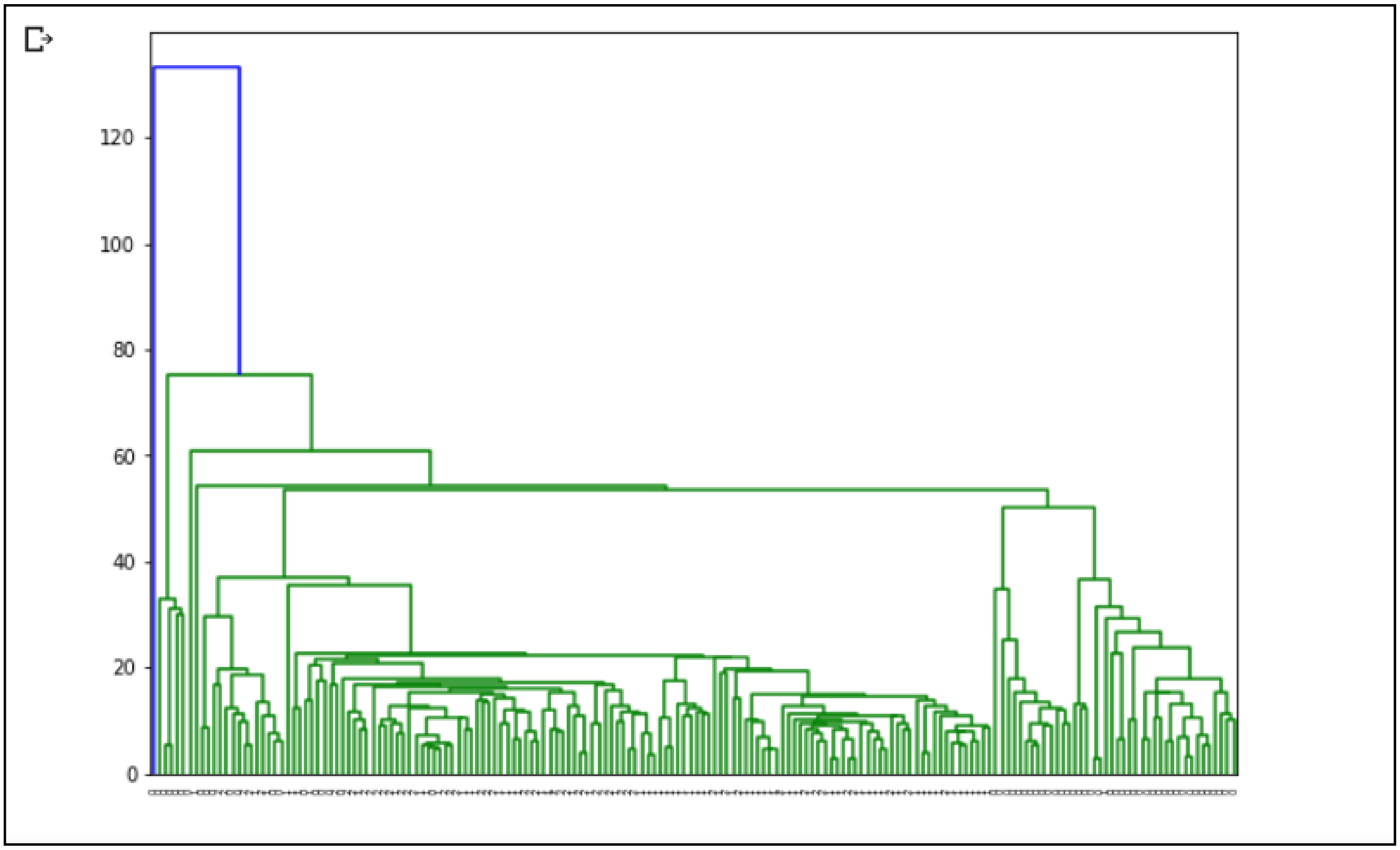
```python
plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

from scipy.cluster.hierarchy import dendrogram, linkage

linked = linkage(x, 'single')
plt.figure(figsize=(10,7))

dendrogram(linked,
           orientation='top',
           labels=iris.target,
           distance_sort='descending',
           show_leaf_counts=True)

plt.show()
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine

wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)
```

```python
from scipy.cluster.hierarchy import dendrogram, linkage

linked = linkage(x, 'single')
plt.figure(figsize=(10,7))

dendrogram(linked,
           orientation='top',
           labels=wine.target,
           distance_sort='descending',
           show_leaf_counts=True)

plt.show()
```



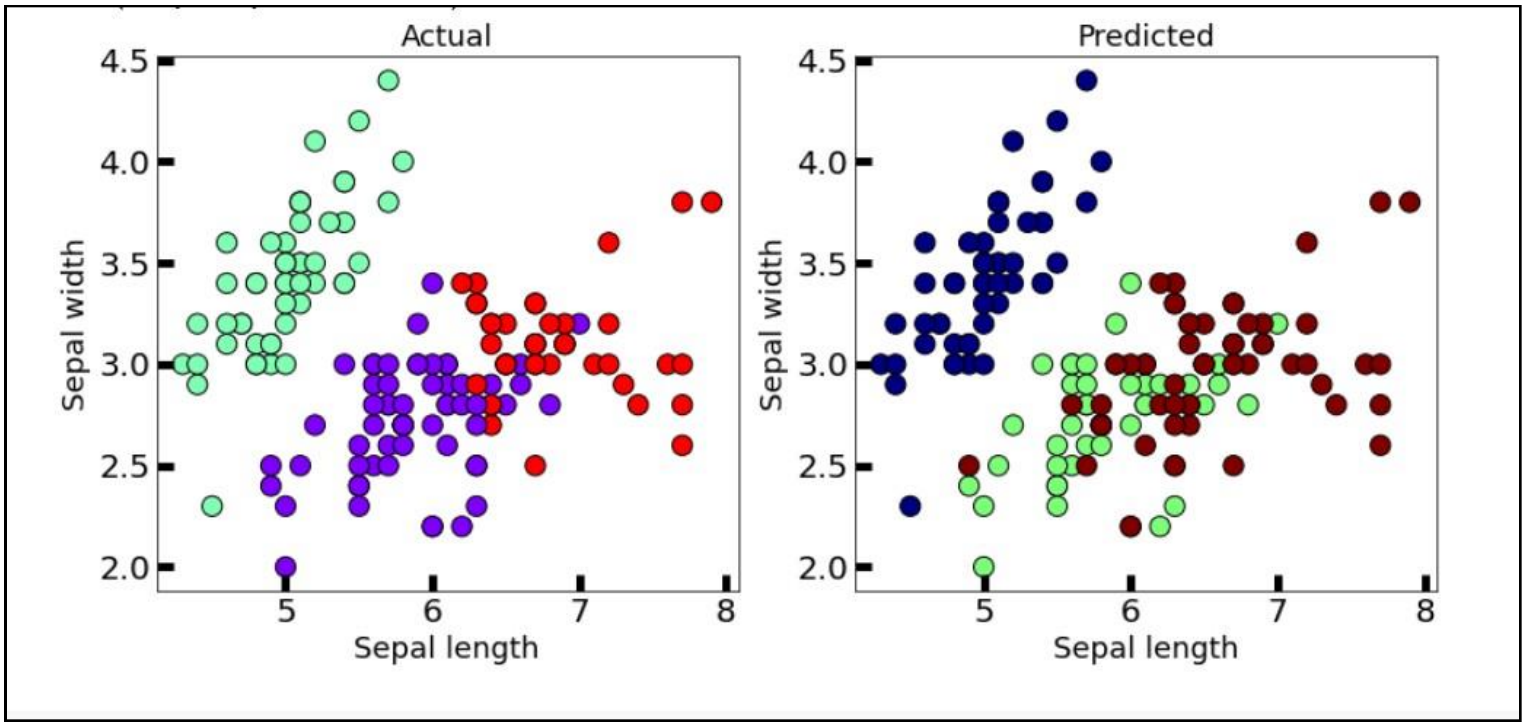
A dendrogram is a diagram that shows the hierarchical relationship between objects. It is most commonly created as an output from hierarchical clustering. The main use of a dendrogram is to work out the best way to allocate objects to clusters.

4) *Hierarchical: AGNES*

IRIS PLANT DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
```

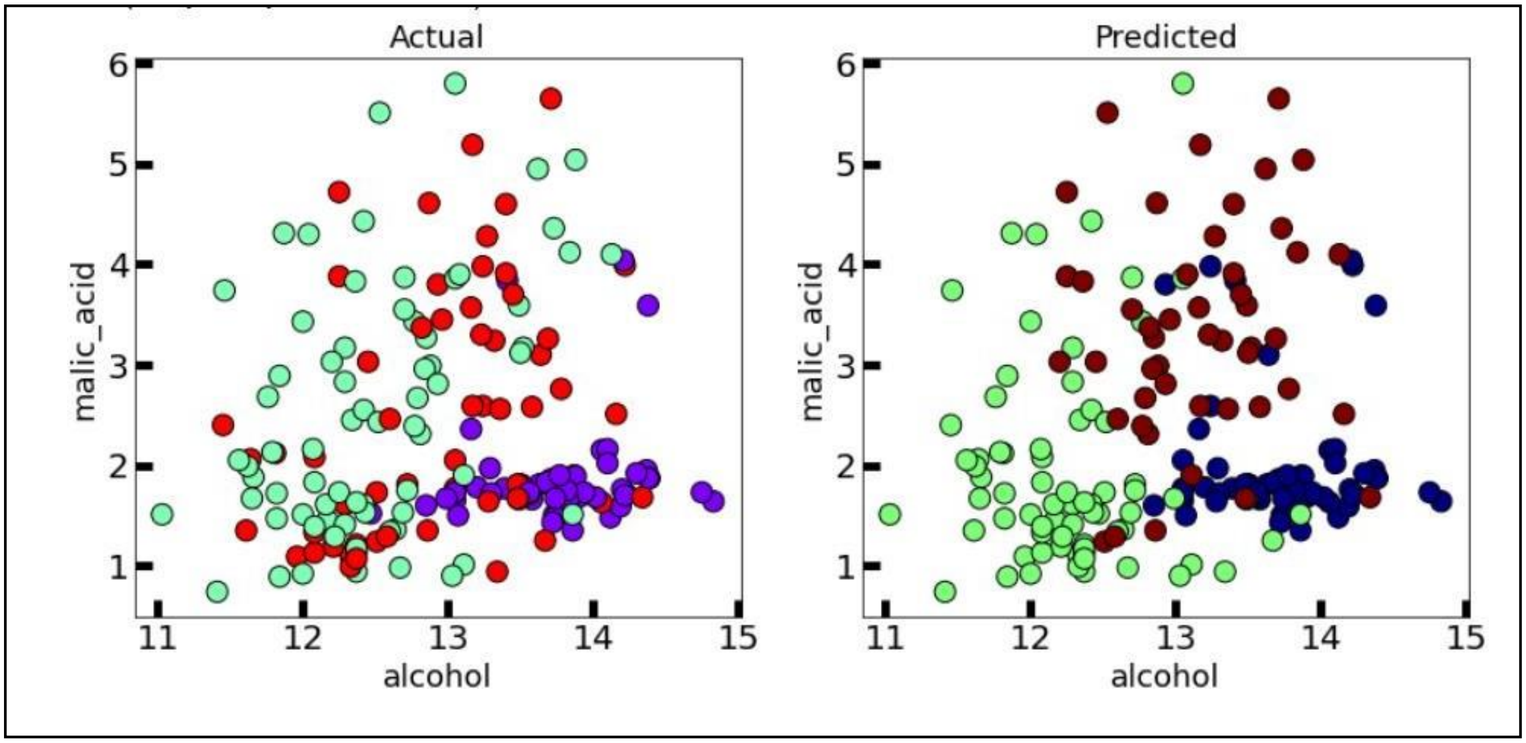
```
df

x=iris.data

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

from sklearn.cluster import AgglomerativeClustering

cluster = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
y = cluster.fit_predict(x)

print("Cluster labels:")
print(cluster.labels_)

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=cluster.labels_, cmap='rainbow') #try using cmap=
'rainbow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine

wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'
```

```
plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)

from sklearn.cluster import AgglomerativeClustering

cluster = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
y = cluster.fit_predict(x)

print("Cluster Labels")
print(cluster.labels_)

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=cluster.labels_, cmap='rainbow') #try using cmap='rainb
ow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



The agglomerative clustering is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It’s also known as AGNES (Agglomerative Nesting).

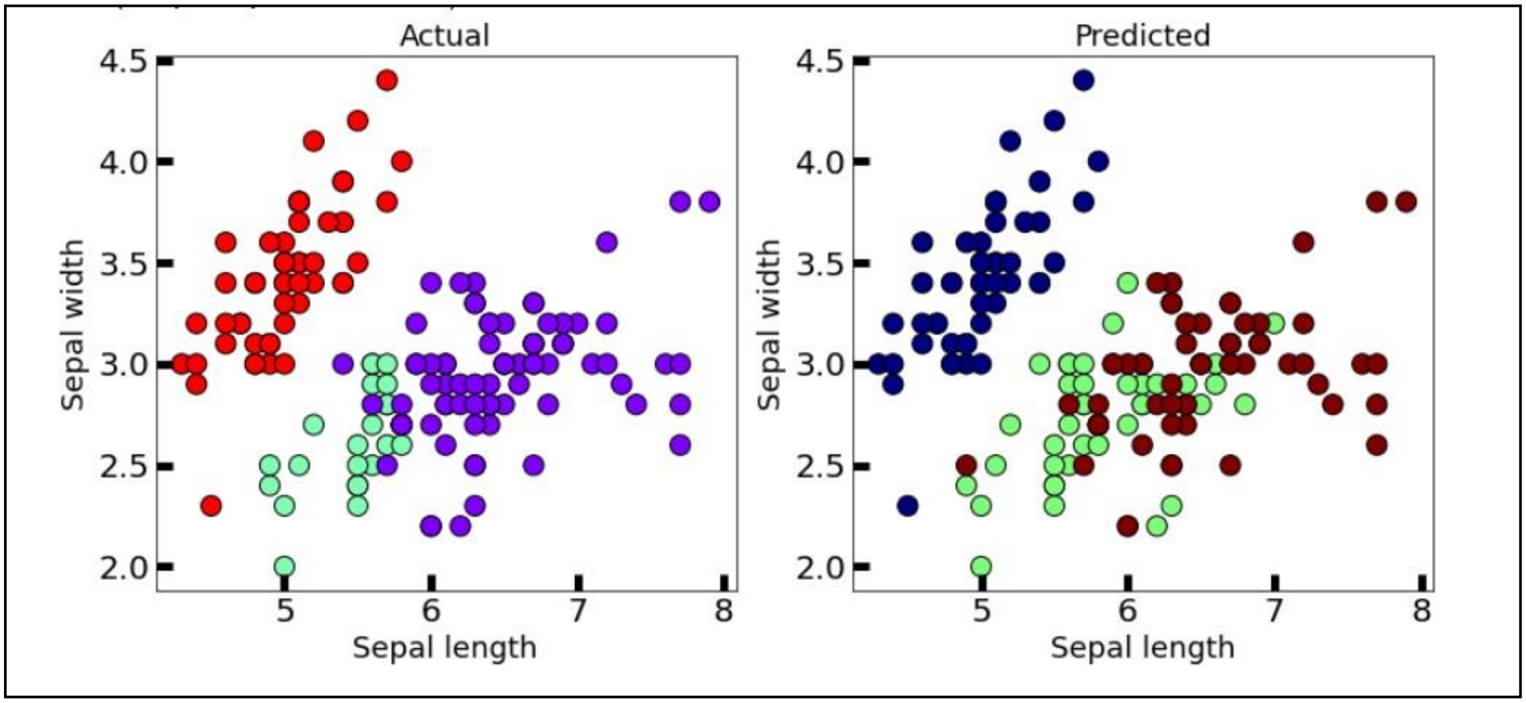
The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects.

The result is a tree-based representation of the objects, named dendrogram.

5) *Hierarchical: BIRCH*

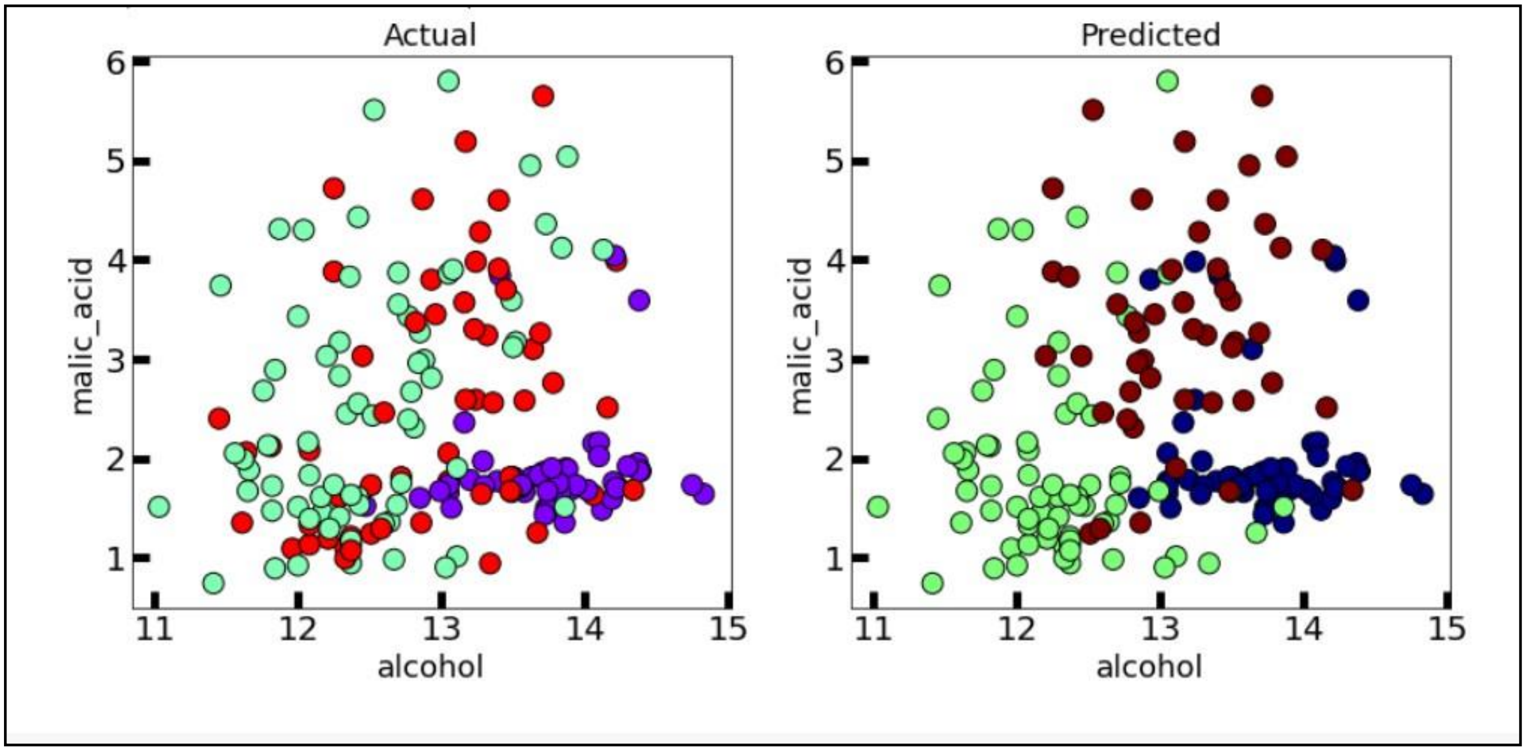
IRIS PLANT DATASET

```
 #importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

from sklearn.cluster import Birch

birch = Birch(n_clusters=3, compute_labels=True, branching_factor=50)
y = birch.fit_predict(x)

print("Cluster Labels")
print(cluster.labels_)

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=birch.labels_, cmap='rainbow') #try using cmap='r
ainbow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine

wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)

from sklearn.cluster import Birch

birch = Birch(n_clusters=3, compute_labels=True, branching_factor=50)
y = birch.fit_predict(x)

print("Cluster Labels")
print(cluster.labels_)

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=birch.labels_, cmap='rainbow') #try using cmap='rainbow
'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) is a clustering algorithm that can cluster large datasets by first generating a small and compact summary of the large dataset that retains as much information as possible.

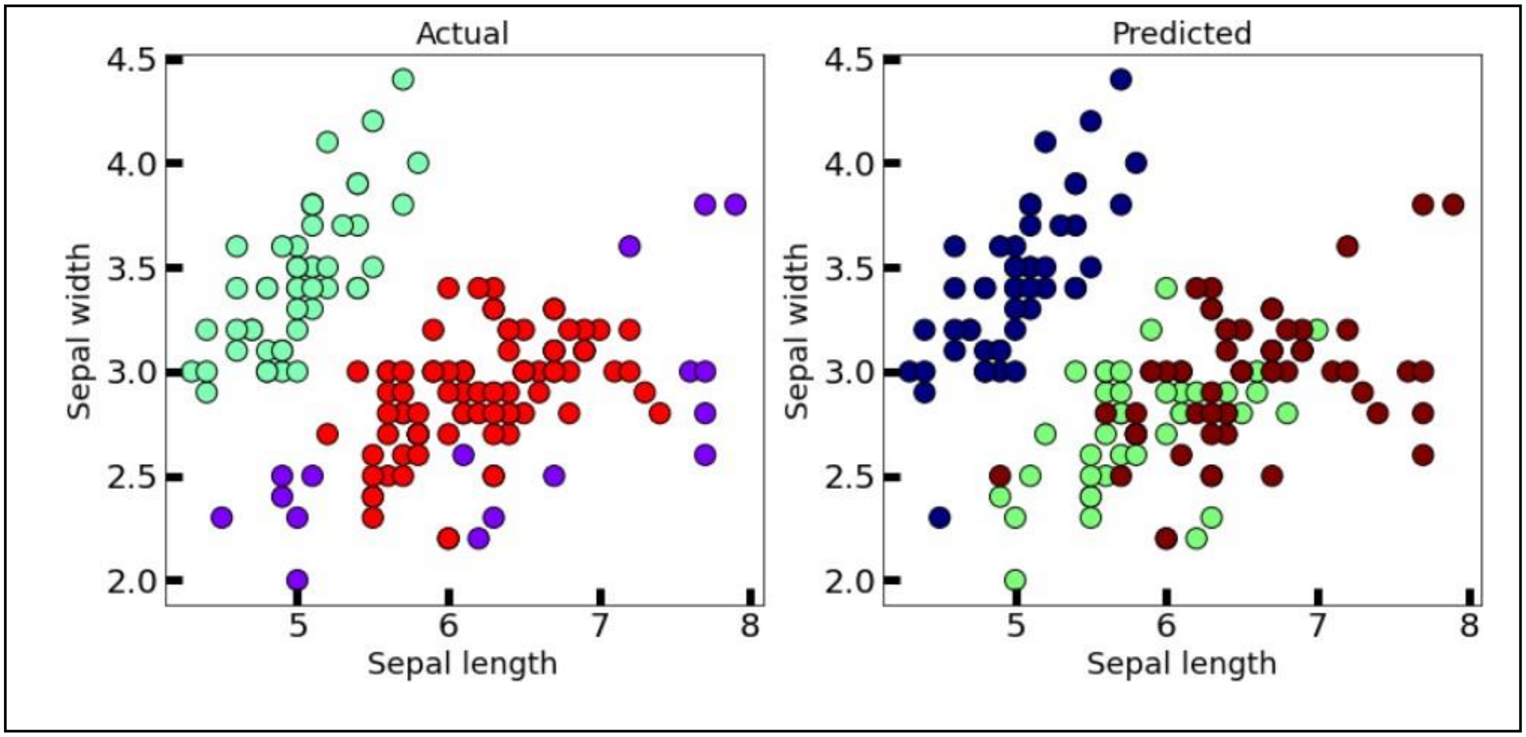
This smaller summary is then clustered instead of clustering the larger dataset

6) *Density based: DBSCAN*

IRIS PLANT DATASET

```
 #importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data
```

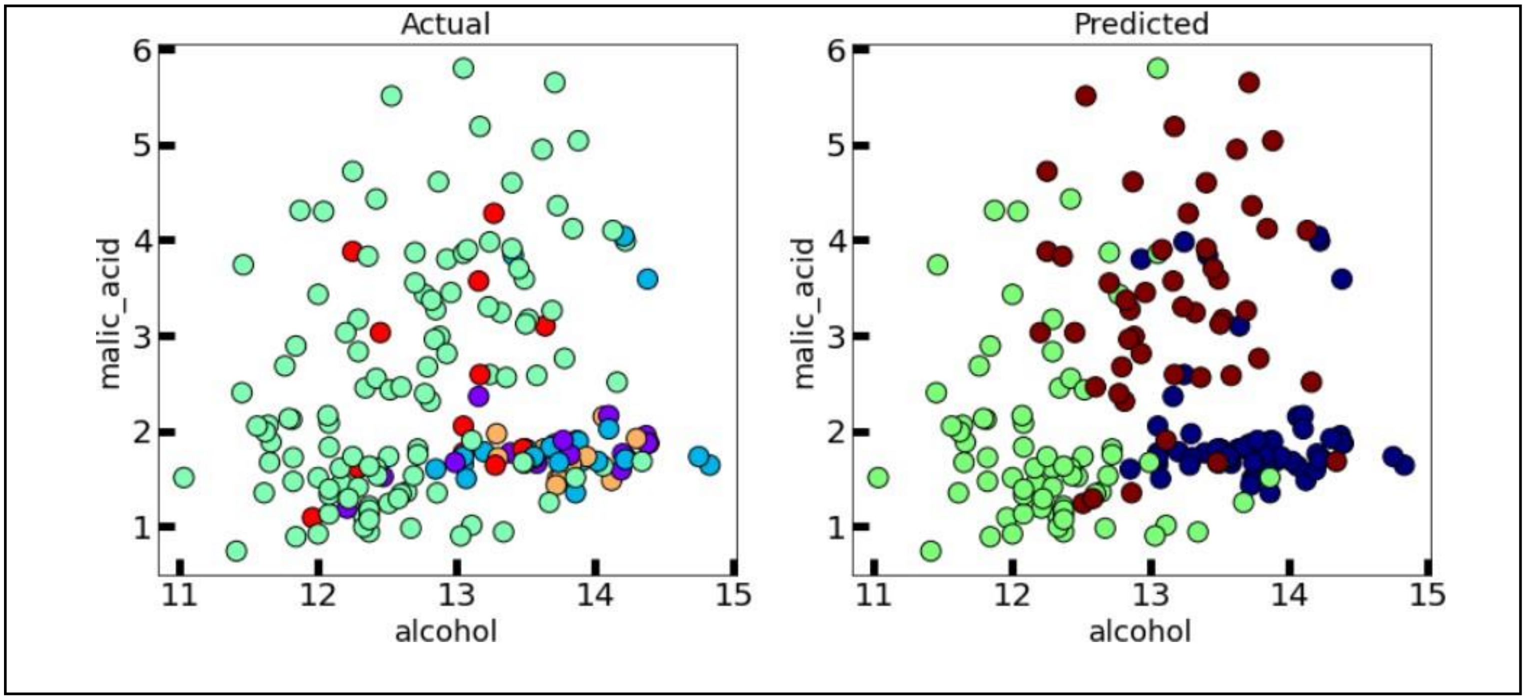
```
plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.5, algorithm='auto', metric='euclidean')
y = dbscan.fit_predict(x)

print("Cluster Labels")
print(dbscan.labels_)

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=dbscan.labels_, cmap='rainbow') #try using cmap='
rainbow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
 #importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine

wine=load_wine()   #loading iris dataset from sklearn.datasets
wine


x=wine.data


df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df
```

```
plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)


from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=35, algorithm='auto', metric='euclidean')
y = dbscan.fit_predict(x)


print("Cluster Labels")
print(dbscan.labels_)


plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=dbscan.labels_, cmap='rainbow') #try using cmap='rainbo
w'
plt.show()


fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



Clusters are dense regions in the data space, separated by regions of the lower density of points. The

DBSCAN algorithm is based on this intuitive notion of “clusters” and “noise”.

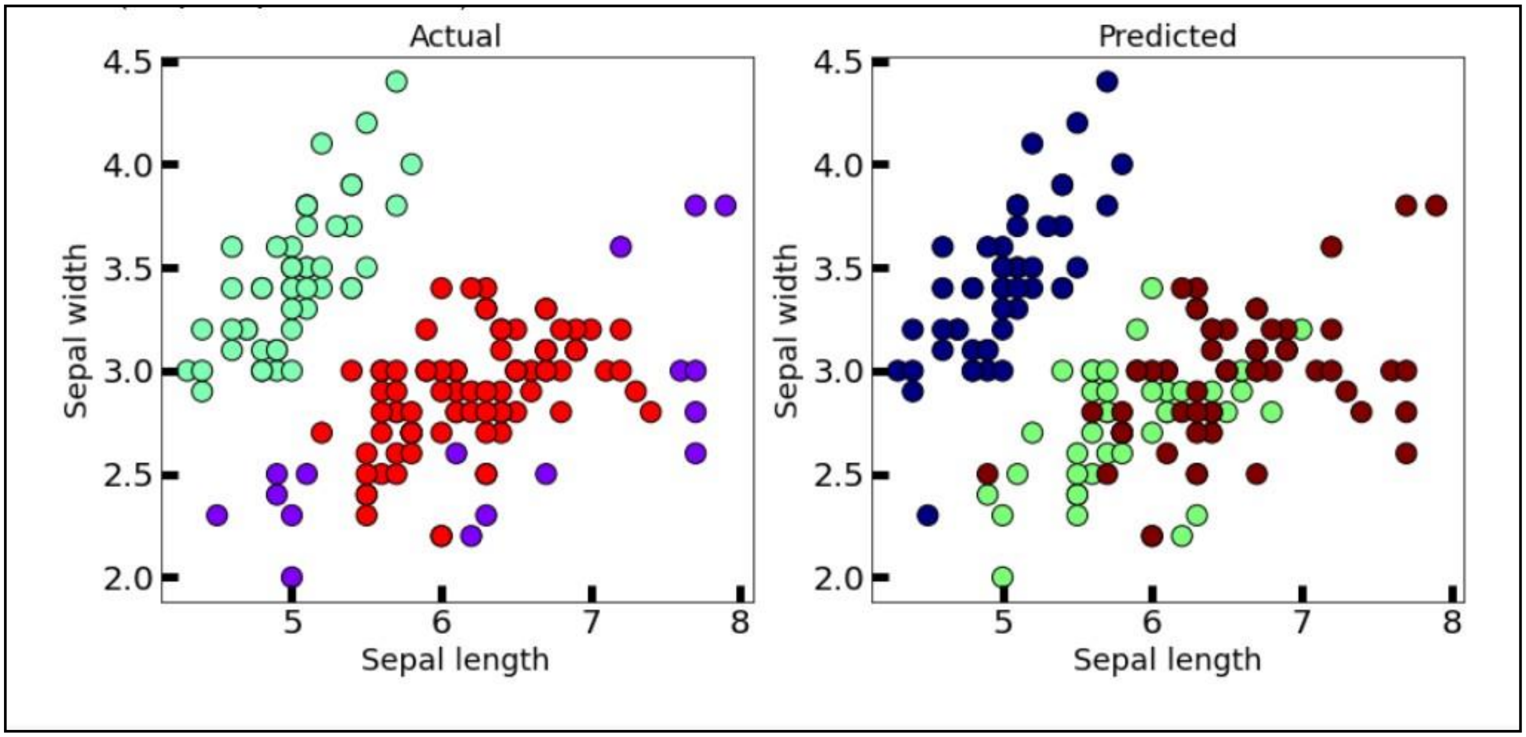
The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.

7) *Density based: OPTICS*

IRIS PLANT DATASET

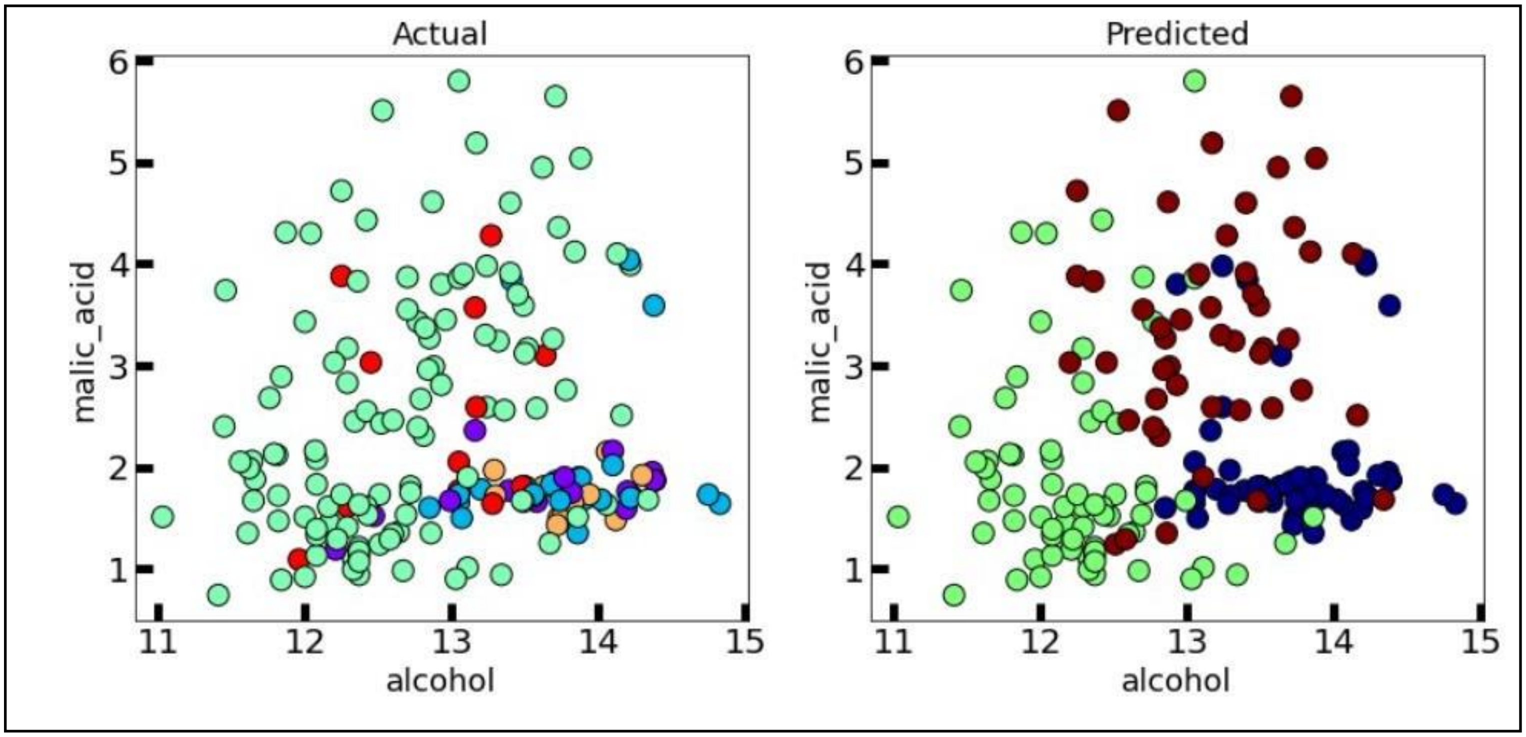
```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.5, algorithm='auto', metric='euclidean')
y = dbscan.fit_predict(x)

print("Cluster Labels")
print(dbscan.labels_)

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=dbscan.labels_, cmap='rainbow') #try using cmap='
rainbow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
```

```
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine

wine=load_wine()    #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=35, algorithm='auto', metric='euclidean')
y = dbscan.fit_predict(x)

print("Cluster Labels")
print(dbscan.labels_)

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=dbscan.labels_, cmap='rainbow') #try using cmap='rainbo
w'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



This clustering technique is different from other clustering techniques in the sense that this technique does not explicitly segment the data into clusters.

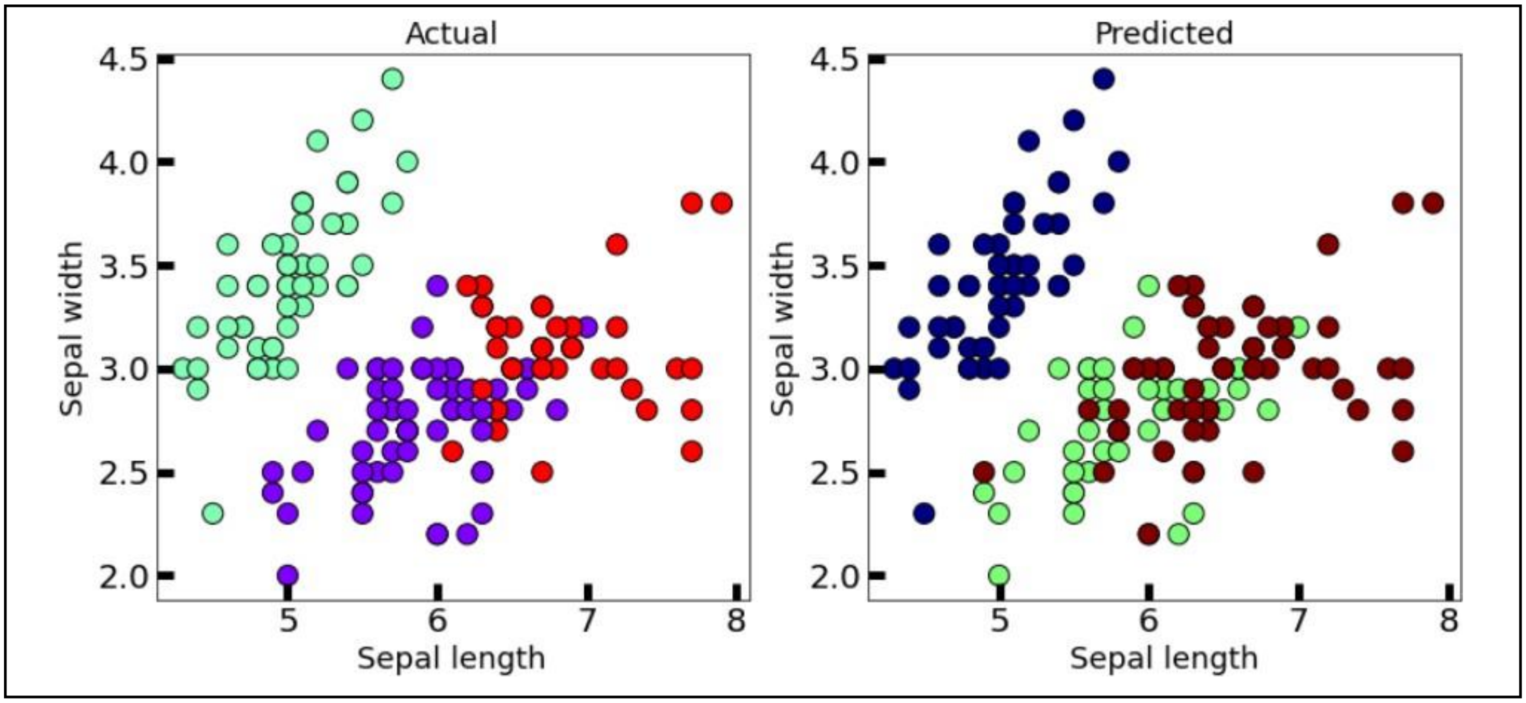
Instead, it produces a visualization of Reachability distances and uses this visualization to cluster the data.

8) *K-means++*

IRIS PLANT DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'
```

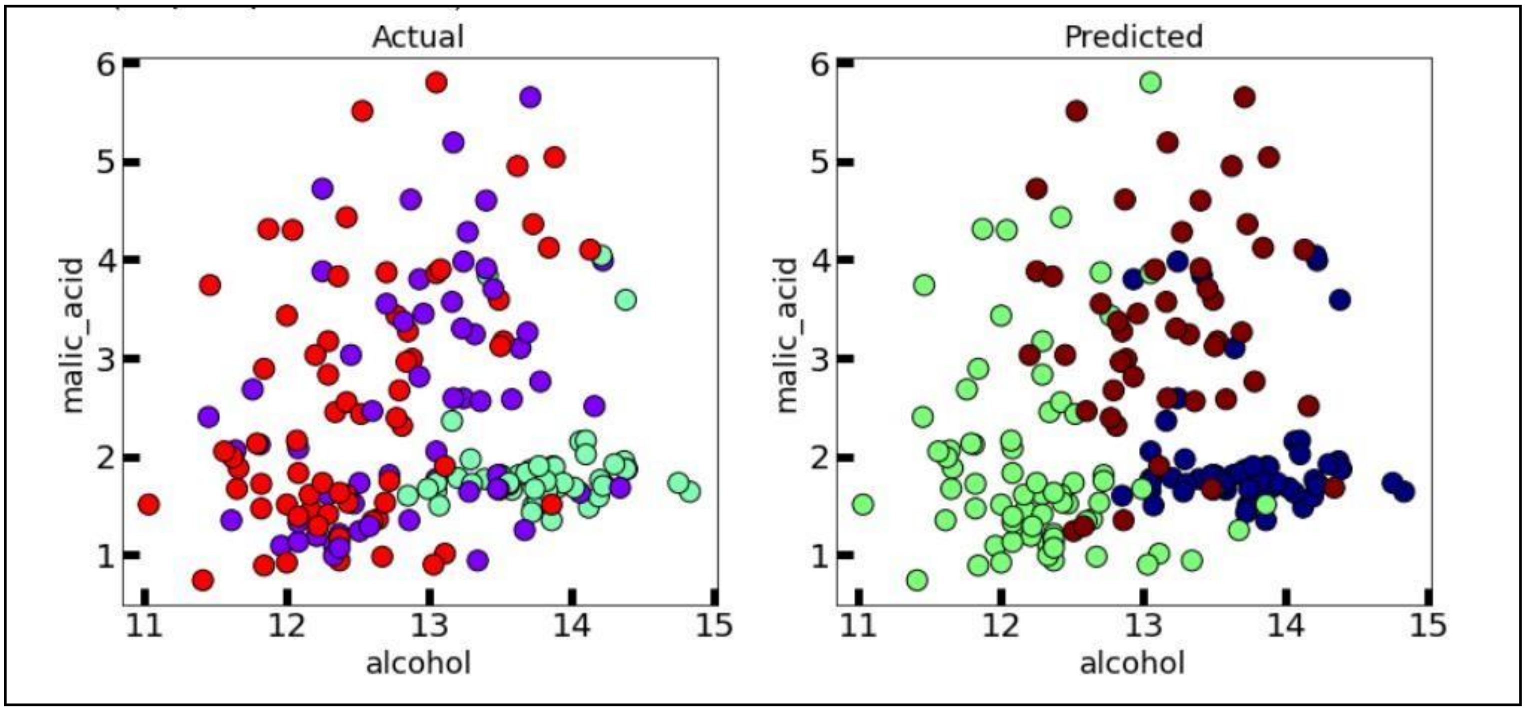
```
plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)

kmeans = KMeans(init='k-means++', n_clusters=3, n_init=10, max_iter=300, random_state=42)
y = kmeans.fit_predict(x)


print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='
rainbow'
plt.show()

fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=y, cmap='rainbow',edgecolor='k', s=150) #you
can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine


wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'
```

```
plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)


kmeans = KMeans(init='k-means++', n_clusters=3, n_init=10, max_iter=300, random_state=42)
y = kmeans.fit_predict(x)


print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)


plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='rainbo
w'
plt.show()


fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=y, cmap='rainbow',edgecolor='k', s=150) #you can al
so try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



In the case of K-Means clustering, we were using randomization. The initial k-centroids were picked randomly from the data points.

This randomization of picking k-centroids points results in the problem of initialization sensitivity. This problem tends to affect the final formed clusters. The final formed clusters depend on how initial centroids were picked.

K-Means++ solves the above problem.

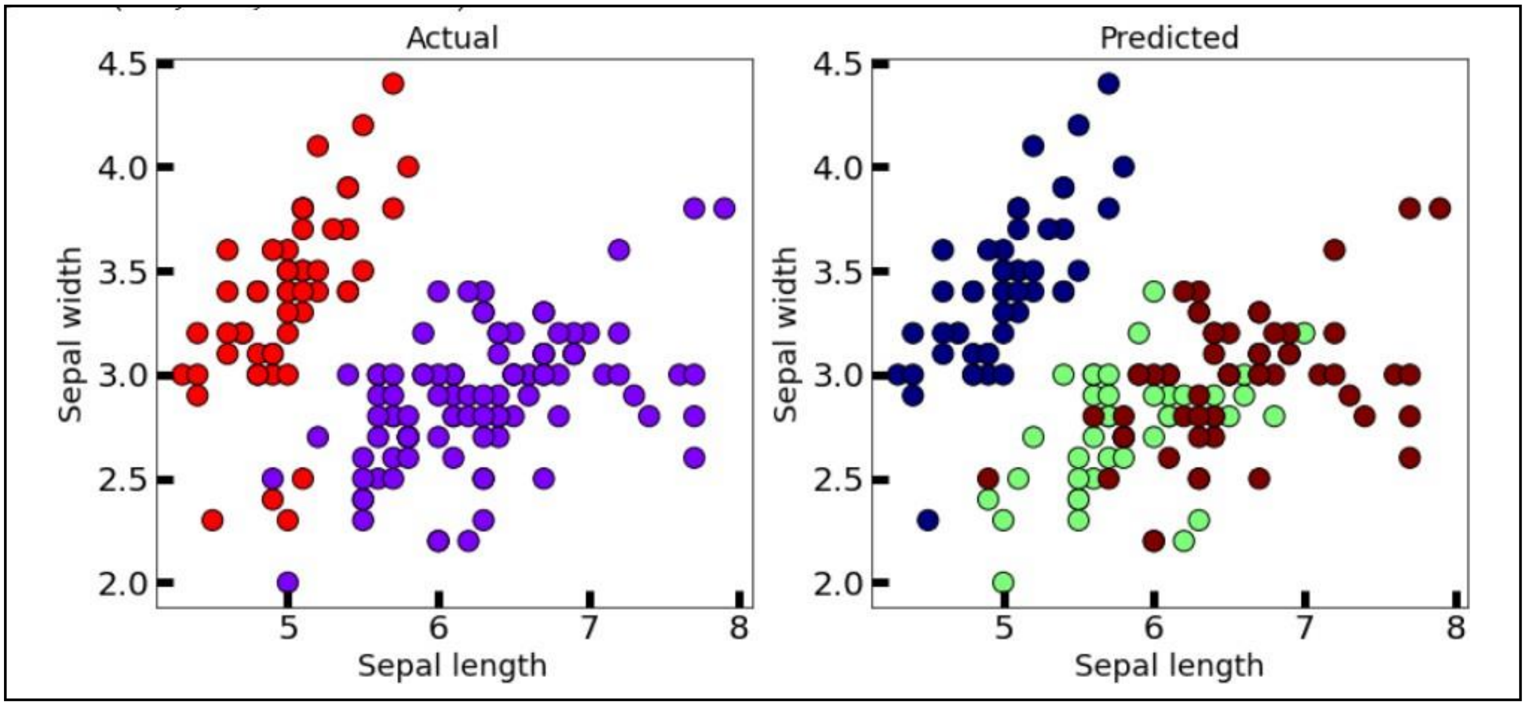
9) *Bisecting K-means*

IRIS PLANT DATASET

```
 #importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
iris=load_iris()   #loading iris dataset from sklearn.datasets
iris

df=pd.DataFrame(data=iris.data, columns=['sepal length','sepal width','petal length','petal width'])
df

x=iris.data

plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=iris.target, cmap='gist_rainbow') #try using cmap
='rainbow'

plt.xlabel('Sepal Width', fontsize=18)
plt.ylabel('Sepal length', fontsize=18)



from sklearn.cluster import KMeans
import numpy as np

K = 2
current_clusters = 1
split = 0
while current_clusters != K:
    kmeans = KMeans(n_clusters=2).fit(x)
    current_clusters += 1
    split += 1
    cluster_centers = kmeans.cluster_centers_
    sse = [0]*2
    for point, label in zip(x, kmeans.labels_):
        sse[label] += np.square(point-cluster_centers[label]).sum()
    chosen_cluster = np.argmax(sse, axis=0)
    chosen_cluster_data = x[kmeans.labels_ == chosen_cluster]
    x = chosen_cluster_data



print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)
```

```
plt.scatter(x=df['sepal length'], y=df['sepal width'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='
rainbow'
plt.show()
```

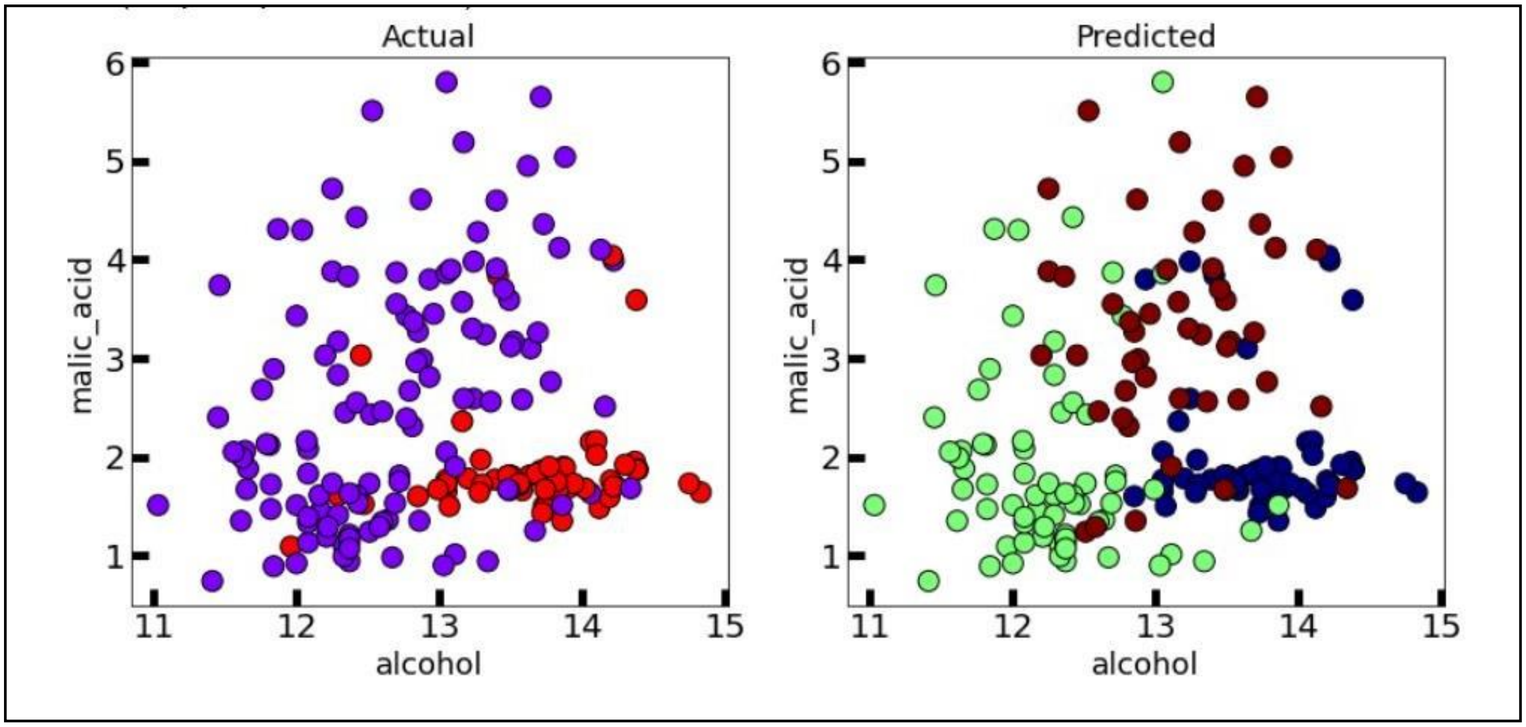
```
fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['sepal length'], y=df['sepal width'], c=kmeans.labels_, cmap='rainbow',edgecolor='k',
 s=150) #you can also try cmap='rainbow'
axes[1].scatter(x=df['sepal length'], y=df['sepal width'], c=iris.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('Sepal length', fontsize=18)
axes[0].set_ylabel('Sepal width', fontsize=18)
axes[1].set_xlabel('Sepal length', fontsize=18)
axes[1].set_ylabel('Sepal width', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



WINE DATASET

```
#importing libraries
import numpy as np
import pandas as pd
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets import load_wine


wine=load_wine()   #loading iris dataset from sklearn.datasets
wine

x=wine.data

df=pd.DataFrame(data=wine.data, columns=['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline'])
df

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=wine.target, cmap='gist_rainbow') #try using cmap='rain
bow'

plt.xlabel('alcohol', fontsize=18)
plt.ylabel('malic_acid', fontsize=18)

from sklearn.cluster import KMeans
import numpy as np

K = 2
current_clusters = 1
split = 0
while current_clusters != K:
    kmeans = KMeans(n_clusters=2).fit(x)
    current_clusters += 1
    split += 1
    cluster_centers = kmeans.cluster_centers_
    sse = [0]*2
    for point, label in zip(x, kmeans.labels_):
        sse[label] += np.square(point-cluster_centers[label]).sum()
    chosen_cluster = np.argmax(sse, axis=0)
    chosen_cluster_data = x[kmeans.labels_ == chosen_cluster]
    x = chosen_cluster_data

print("K-Means Cluster Centers")
print(kmeans.cluster_centers_)
print("Cluster Labels")
print(kmeans.labels_)

plt.scatter(x=df['alcohol'], y=df['malic_acid'] ,c=kmeans.labels_, cmap='rainbow') #try using cmap='rainbo
w'
plt.show()
```

```
fig, axes = plt.subplots(1, 2, figsize=(14,6))
axes[0].scatter(x=df['alcohol'], y=df['malic_acid'], c=kmeans.labels_, cmap='rainbow',edgecolor='k', s=150
) #you can also try cmap='rainbow'
axes[1].scatter(x=df['alcohol'], y=df['malic_acid'], c=wine.target, cmap='jet',edgecolor='k', s=150)
axes[0].set_xlabel('alcohol', fontsize=18)
axes[0].set_ylabel('malic_acid', fontsize=18)
axes[1].set_xlabel('alcohol', fontsize=18)
axes[1].set_ylabel('malic_acid', fontsize=18)
axes[0].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[1].tick_params(direction='in', length=10, width=5, colors='k', labelsize=20)
axes[0].set_title('Actual', fontsize=18)
axes[1].set_title('Predicted', fontsize=18)
```



Bisecting K-means clustering technique is a little modification to the regular K-Means algorithm, wherein we can fix the procedure of dividing the data into clusters.

So, similar to K-means, we first initialize K centroids (You can either do this randomly or can have some prior).

After which we apply regular K-means with K=2 (that's why the word bisecting). We keep repeating this bisection step until the desired number of clusters are reached.

For the Silhouette Score , Calinski Harabasz Score , Davies Bouldin Score , SSE and SSB I have written the code in only the first parts as it is same for all the comparisons and the comparison table is as shown :

Type of Algorithm	**Algorithm**	**Dataset**	**Silhouette Score**	**Calinski Harabasz Score**	**Davies Bouldin Score**	**SSE**	**SSB**
Partition Based	K-means	IRIS PLANT DATASET	0.5528190124	561.6277566	0.6619715465	78.85144143	24.18035247
		WINE DATASET	0.5711381938	561.8156579	0.5342431775	2370689.687	-2370571.805
	K-medoids	IRIS PLANT DATASET	0.5201984013	521.5609065	0.668624441	98.86857318	5.11476015
		WINE	0.566648040	539.3792354	0.529239412	16376.969	-

		DATASET	9		6	32	16243.64278
Hierarchical	Dendrogram	IRIS PLANT DATASET	-	-	-	-	-
		WINE DATASET	-	-	-	-	-
	AGNES	IRIS PLANT DATASET	0.5543236611	558.0580408	558.0580408	-	-
		WINE DATASET	0.5644796402	552.8517115	0.5357343074	-	-
	BIRCH	IRIS PLANT DATASET	0.5019524848	458.4725106	0.6258305924	-	-
		WINE DATASET	0.5644796402	552.8517115	0.5357343074	-	-
Density Based	DBSCAN	IRIS PLANT DATASET	0.486034197	220.297515	7.222448016	-	-
		WINE DATASET	0.4413295945	208.9449396	7.812129203	-	-
	OPTICS	IRIS PLANT DATASET	0.486034197	220.297515	7.222448016	-	-
		WINE DATASET	0.4413295945	208.9449396	7.812129203	-	-
Additional	K-means++	IRIS PLANT DATASET	0.5528190124	561.6277566	0.6619715465	78.85144143	24.18035247
		WINE DATASET	0.5711381938	561.8156579	0.5342431775	2370689.687	-2370571.805
	Bisecting K-means	IRIS PLANT DATASET	0.3093066205	61.17725176	1.099971025	152.3479518	-44.7653206
		WINE DATASET	0.00384025695	1.06619667	9.045634695	4543749.615	-4543626.929