Task 2 - The Sparks Foundation

Name- Smita Rautmare

Prediction Using Unsupervised ML

From the given 'Iris' dataset, predict the optimum number of clusters and represent it visually.

In [*]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [ ]:
```
import warnings
warnings.filterwarnings('ignore')`
```

In [5]:
```
from sklearn.cluster import KMeans
```

In [10]:
```
data = pd.read_csv('./iris (1).csv')
```

In [11]:
```
data.head()
```

Out[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [12]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
Id               150 non-null int64
SepalLengthCm    150 non-null float64
SepalWidthCm     150 non-null float64
PetalLengthCm    150 non-null float64
PetalWidthCm     150 non-null float64
Species          150 non-null object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [13]:
```
data.describe()
```

Out[13]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [19]:
```
data['Species'].unique()
```

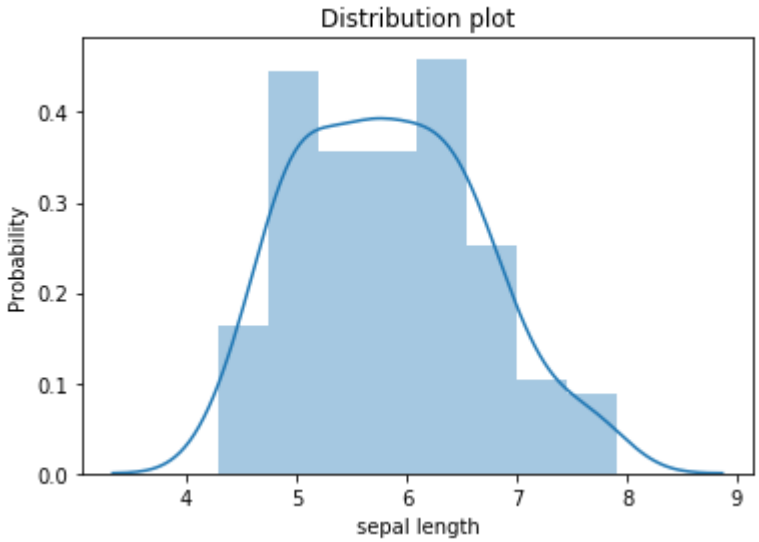
Out[19]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [25]:
```
c =data.corr()
c
```

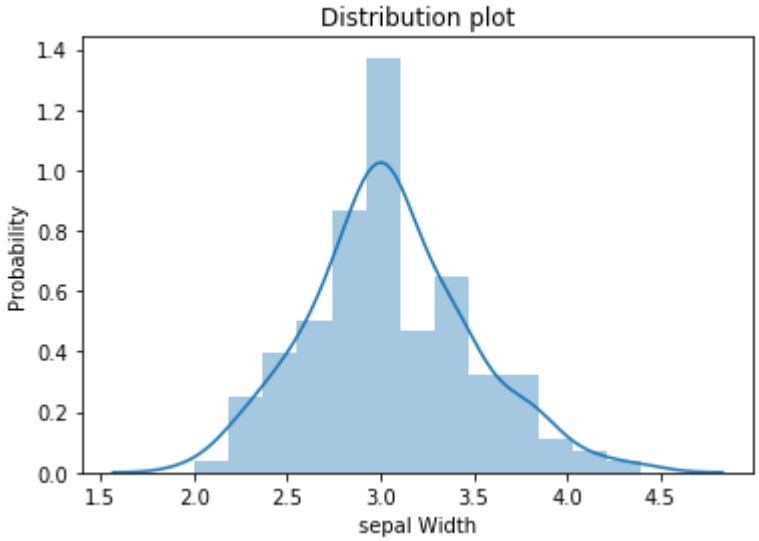
Out[25]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Id	1.000000	0.716676	-0.397729	0.882747	0.899759
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000

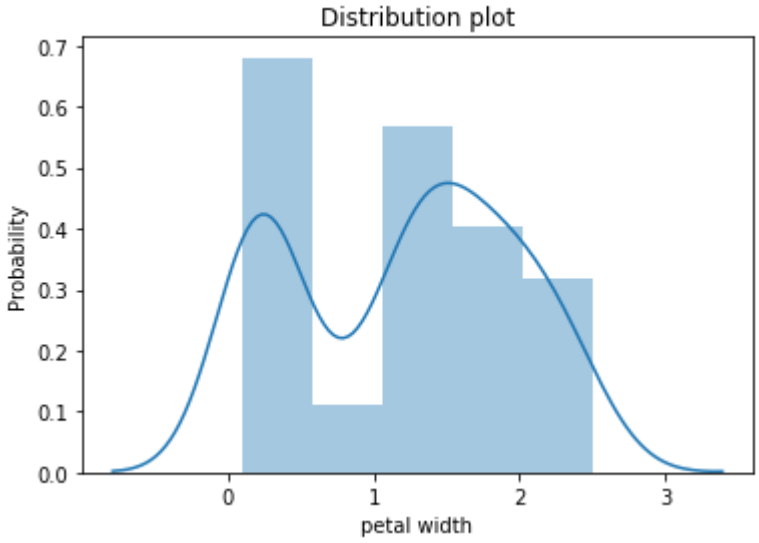
In [39]:
```
sns.distplot(data['SepalLengthCm'])
plt.xlabel('sepal length')
plt.ylabel('Probability')
plt.title('Distribution plot')
plt.show()
```



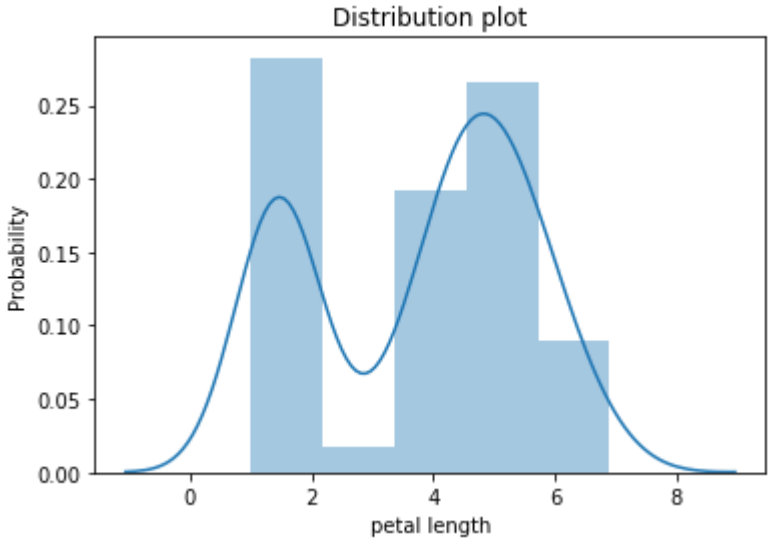
In [41]:
```
sns.distplot(data['SepalWidthCm'])
plt.xlabel('sepal Width')
plt.ylabel('Probability')
plt.title('Distribution plot')
plt.show()
```



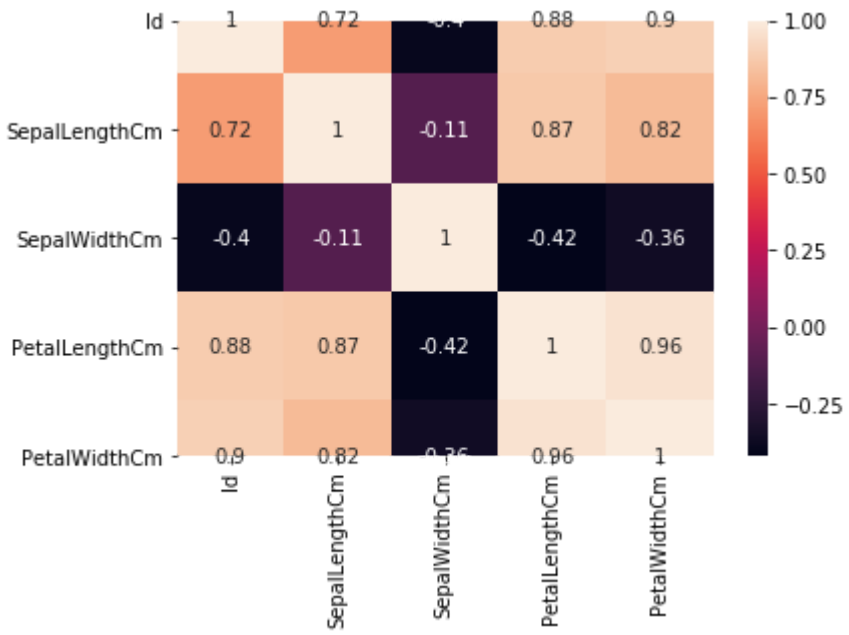
In [40]:
```
sns.distplot(data['PetalWidthCm'])
plt.xlabel('petal width')
plt.ylabel('Probability')
plt.title('Distribution plot')
plt.show()
```



In [42]:
```
sns.distplot(data['PetalLengthCm'])
plt.xlabel('petal length')
plt.ylabel('Probability')
plt.title('Distribution plot')
plt.show()
```



In [30]:
```
c=sns.heatmap(c,annot=True)
plt.show()
```



K means clustering

In [55]:
```
kl=KMeans(n_clusters=3)
```

In [56]:
```
predict=kl.fit_predict(data[['SepalLengthCm','SepalWidthCm']])
predict
```

Out[56]:
```
array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0])
```

In [57]:
```
data['Cluster']=predict
```

In [58]:
```
# added column
data.head()
```

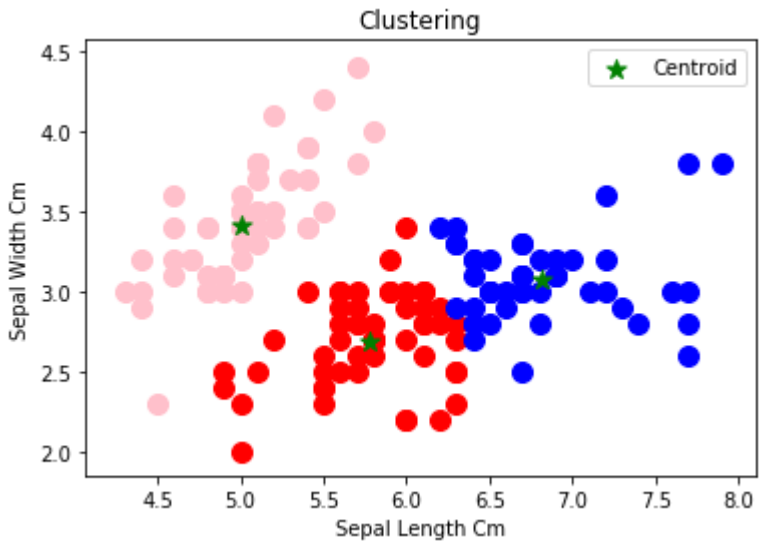
Out[58]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	Cluster
0	1	5.1	3.5	1.4	0.2	Iris-setosa	2
1	2	4.9	3.0	1.4	0.2	Iris-setosa	2
2	3	4.7	3.2	1.3	0.2	Iris-setosa	2
3	4	4.6	3.1	1.5	0.2	Iris-setosa	2
4	5	5.0	3.6	1.4	0.2	Iris-setosa	2

In [60]:
```
centroid=kl.cluster_centers_
```

In [65]:
```
c1 = data[data.Cluster==0]
c2 = data[data.Cluster==1]
c3 = data[data.Cluster==2]
plt.scatter(c1['SepalLengthCm'],c1['SepalWidthCm'],color="red",s=100)
plt.scatter(c2['SepalLengthCm'],c2['SepalWidthCm'],color="blue",s=100)
plt.scatter(c3['SepalLengthCm'],c3['SepalWidthCm'],color="pink",s=100)
plt.scatter(centroid[:,0],centroid[:,1],color='green',marker='*', label='Centroi
plt.xlabel('Sepal Length Cm')
plt.ylabel('Sepal Width Cm')
plt.title('Clustering')
plt.legend()
```

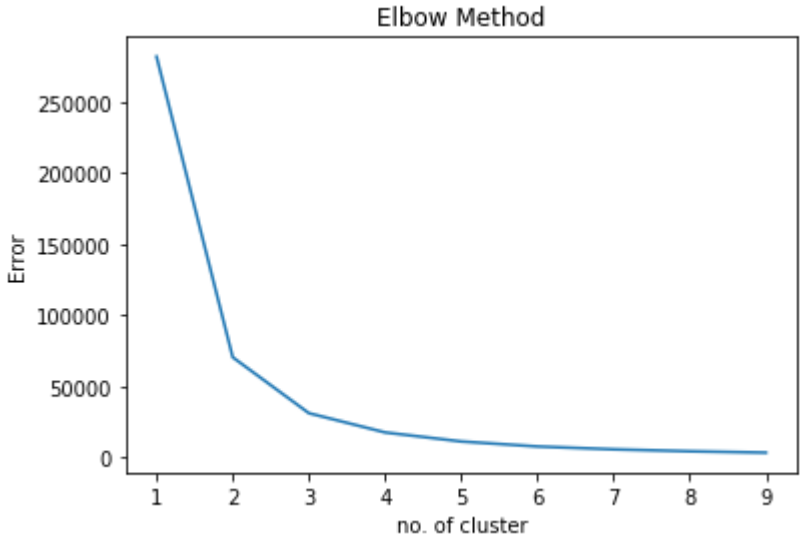
Out[65]: <matplotlib.legend.Legend at 0x2b63e895c88>



here we use elbow method to determine no. of cluster inside the data

In [71]:
```
a=[]
for i in range(1,10):
    kl=KMeans(n_clusters=i).fit(X)
    kl.fit(X)
    a.append(kl.inertia_)
```

In [72]:
```
plt.plot(range(1,10),a)
plt.title('Elbow Method')
plt.xlabel('no. of cluster')
plt.ylabel('Error')
plt.show()
```



Colclusion : K_ means clustering is used to find groups in data ,with the number of groups represented by variable K and data points are clustered based on feature similarity

In [ ]:
```
```