In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [4]:
```
df=pd.read_csv("iris.csv",header=None,names=['s_length',"s_width","p_length","p_w
```

In [5]:
```
df.head()
```

Out[5]:

	s_length	s_width	p_length	p_width	target
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

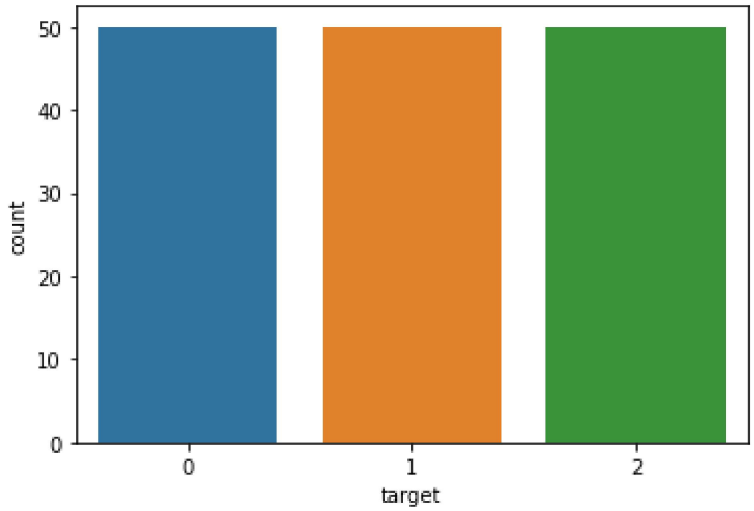
In [6]:
```
df.describe()
```

Out[6]:

	s_length	s_width	p_length	p_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

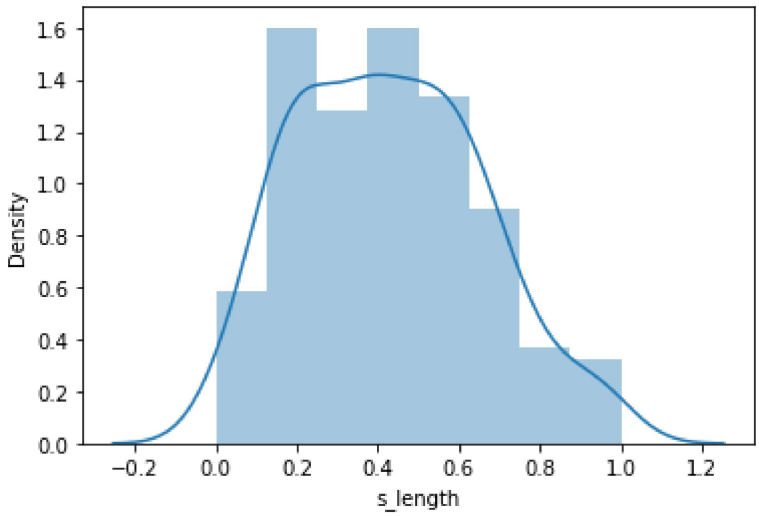
In [46]:
```
sns.countplot(df['target'])
```

Out[46]:
```
<AxesSubplot:xlabel='target', ylabel='count'>
```



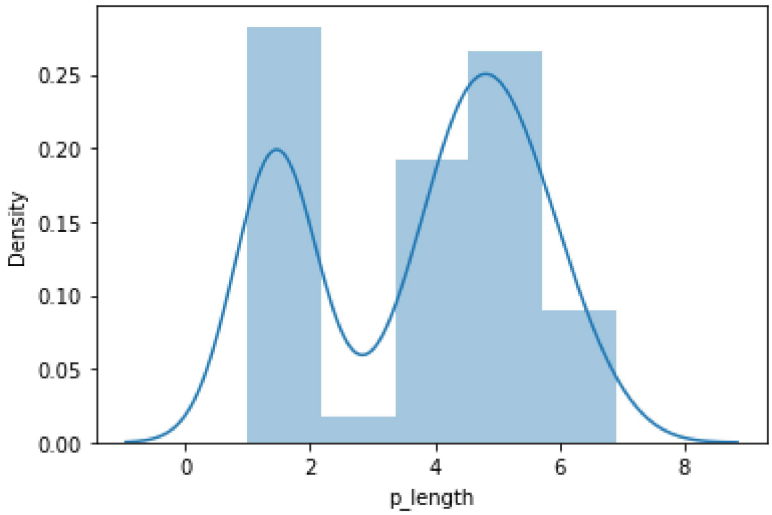
In [47]:
```
sns.distplot(df['s_length'])
```

Out[47]:
```
<AxesSubplot:xlabel='s_length', ylabel='Density'>
```



In [48]:
```
sns.distplot(df['p_length'])
```

Out[48]: <AxesSubplot:xlabel='p_length', ylabel='Density'>



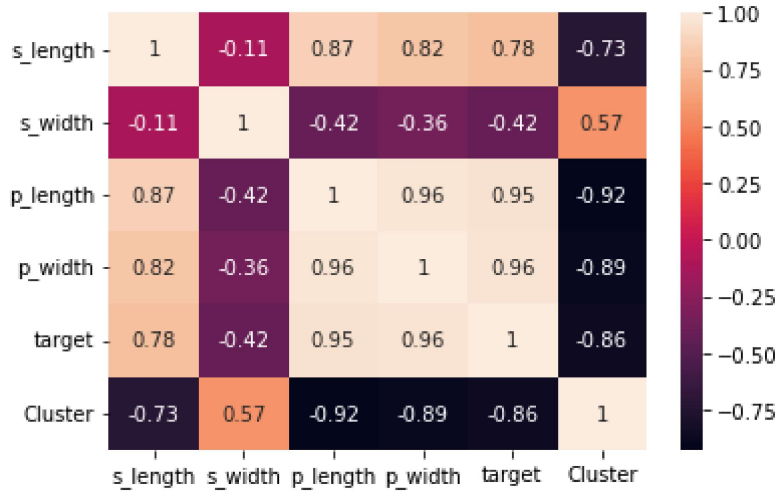
In [7]:
```
df.isnull().sum()
```

Out[7]:
```
s_length    0
s_width     0
p_length    0
p_width     0
target      0
dtype: int64
```

In [28]:
```
corr=df.corr()
```

In [29]:
```
sns.heatmap(corr,annot=True)
```

Out[29]: <AxesSubplot:>



In [ ]:

```
```

In [8]:

```
df['target'].unique()
```

Out[8]:

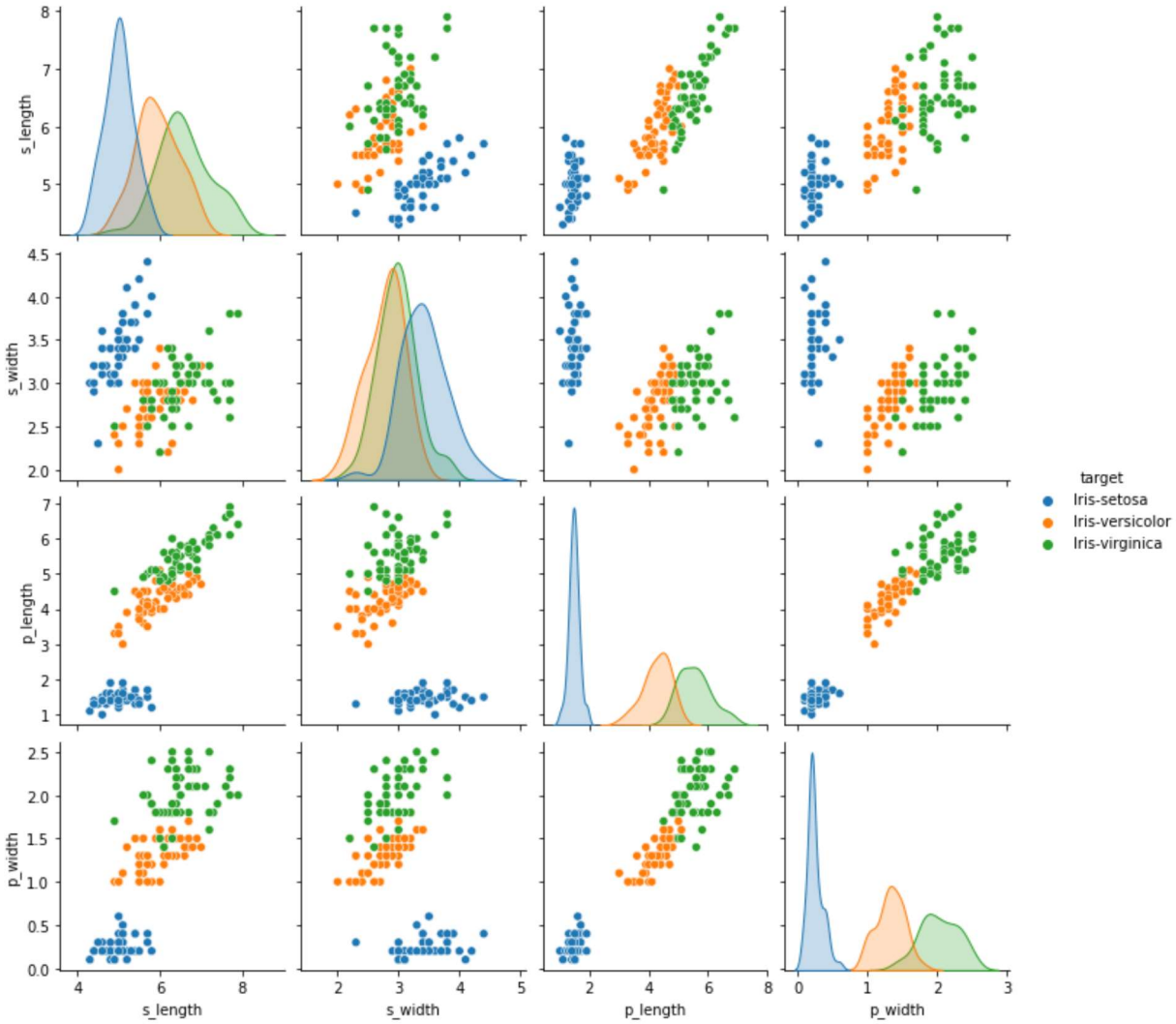
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [9]:

```
sns.pairplot(df,hue="target")
```

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x1c307276670>
```



In [10]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['target']=le.fit_transform(df['target'])
```

In [12]:
```
df
```

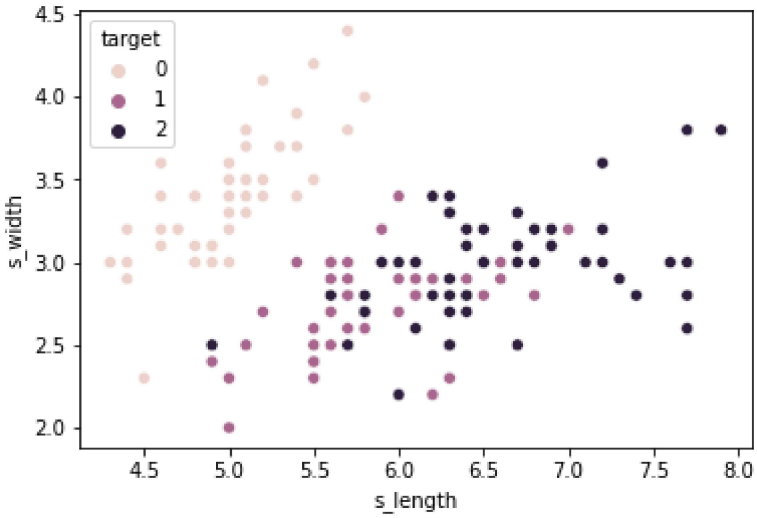
Out[12]:

	s_length	s_width	p_length	p_width	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

In [18]:
```
sns.scatterplot(df['s_length'],df['s_width'],hue=df['target'])
```

Out[18]: <AxesSubplot:xlabel='s_length', ylabel='s_width'>

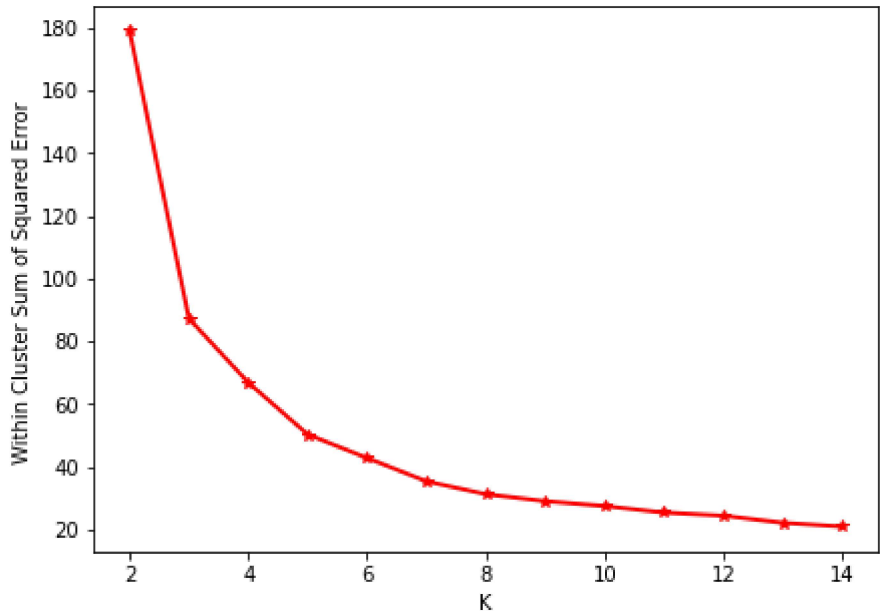


In [14]:
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
wcss=[]
sil_score={}
k_range=range(2,15)
for k in k_range:
    km=KMeans(n_clusters=k)
    km.fit(df)
    wcss.append(km.inertia_)
    sil_score[k]=silhouette_score(df,km.labels_,metric="euclidean")
```

In [15]:
```
sil_score
```

Out[15]:
```
{2: 0.6798027978812322,
 3: 0.5816937070990904,
 4: 0.5384620014392415,
 5: 0.5172316508697471,
 6: 0.3967348232035191,
 7: 0.3900178924536382,
 8: 0.37139498092679074,
 9: 0.3632164158089461,
 10: 0.33173921526530836,
 11: 0.31083855776721,
 12: 0.29223851819296515,
 13: 0.33014826961357197,
 14: 0.28468925260472316}
```

In [16]:
```
#Elbow method
plt.figure(figsize=(7,5))
plt.plot(k_range,wcss,marker="*",color="red",linewidth=2)
plt.xlabel("K")
plt.ylabel("Within Cluster Sum of Squared Error ")
plt.show()
```



In [19]:
```
#Clustering with k=6
model=KMeans(n_clusters=3)
y_pred=model.fit_predict(df)
y_pred
```

Out[19]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 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, 2, 2, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

In [20]:
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
wcss=[]
sil_score={}
k_range=range(2,15)
for k in k_range:
    km=KMeans(n_clusters=k)
    km.fit(df)
    wcss.append(km.inertia_)
    sil_score[k]=silhouette_score(df,km.labels_,metric="euclidean")
```

In [21]:
```
sil_score
```

Out[21]:
```
{2: 0.6798027978812322,
 3: 0.5816937070990904,
 4: 0.5384620014392415,
 5: 0.5187741145336747,
 6: 0.3984346314538225,
 7: 0.3905441672121113,
 8: 0.3775296169222094,
 9: 0.37139776692298077,
 10: 0.36194756225152275,
 11: 0.2944235694912579,
 12: 0.3120444843047372,
 13: 0.2857444050595984,
 14: 0.27977379531888735}
```

In [22]:
```
#Clustering with k=6
model=KMeans(n_clusters=2)
y_pred=model.fit_predict(df)
y_pred
```

Out[22]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

In [23]:
```
df['Cluster']=y_pred
```

In [24]:
```
df.head()
```

Out[24]:

	s_length	s_width	p_length	p_width	target	Cluster
0	5.1	3.5	1.4	0.2	0	1
1	4.9	3.0	1.4	0.2	0	1
2	4.7	3.2	1.3	0.2	0	1
3	4.6	3.1	1.5	0.2	0	1
4	5.0	3.6	1.4	0.2	0	1

In [25]:
```
model.cluster_centers_
```

Out[25]:
```
array([[6.27373737, 2.87575758, 4.92525253, 1.68181818, 1.50505051],
       [5.00784314, 3.4       , 1.49411765, 0.26078431, 0.01960784]])
```

In [30]:
```
X= df[['s_length','s_width']]
```

In [31]:
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
wcss=[]
sil_score={}
k_range=range(2,15)
for k in k_range:
    km=KMeans(n_clusters=k)
    km.fit(X)
    wcss.append(km.inertia_)
    sil_score[k]=silhouette_score(X,km.labels_,metric="euclidean")
```

In [32]:
```
sil_score
```

Out[32]:
```
{2: 0.4635854719215507,
 3: 0.4434693231245126,
 4: 0.4206056547103897,
 5: 0.4059857381060708,
 6: 0.4000768016964755,
 7: 0.39640094983182655,
 8: 0.3937570775007619,
 9: 0.3935896186809934,
 10: 0.3950916166164115,
 11: 0.3898686956569604,
 12: 0.4054926847954058,
 13: 0.39308641740401434,
 14: 0.37766127321450493}
```

In [33]:
```
from sklearn.cluster import KMeans
```

In [34]:
```
km=KMeans(n_clusters=2)
y_pred=km.fit_predict(X)
y_pred
```

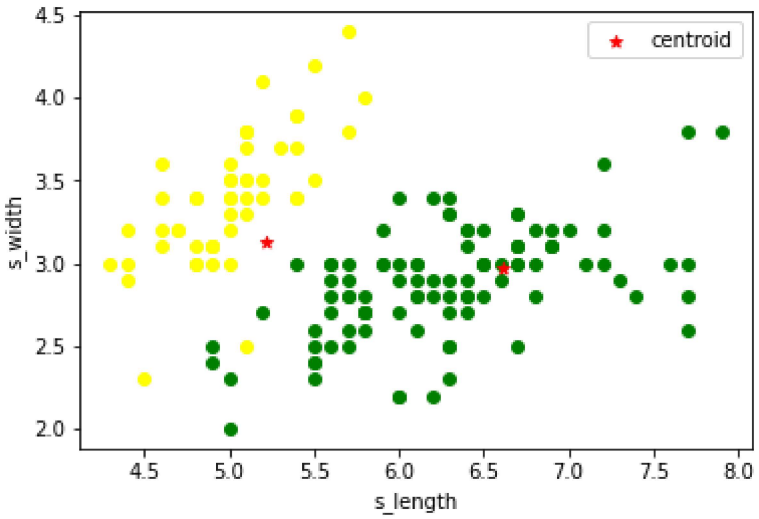
Out[34]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1])
```

In [35]:
```
df1=df[df['Cluster']==0]
df2=df[df['Cluster']==1]
```

In [36]:
```
plt.scatter(df1['s_length'],df1['s_width'],color='green')
plt.scatter(df2['s_length'],df2['s_width'],color="yellow")

plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='red',marker=
plt.xlabel("s_length")
plt.ylabel("s_width")
plt.legend()
```

Out[36]:
```
<matplotlib.legend.Legend at 0x1c308120340>
```



In [37]:
```
km.cluster_centers_
```

Out[37]:
```
array([[6.61044776, 2.96567164],
       [5.22409639, 3.1253012 ]])
```

In [38]:
```
from sklearn.preprocessing import MinMaxScaler
sc=MinMaxScaler()
df['s_length']=sc.fit_transform(df[['s_length']])
df['s_width']=sc.fit_transform(df[['s_width']])
```

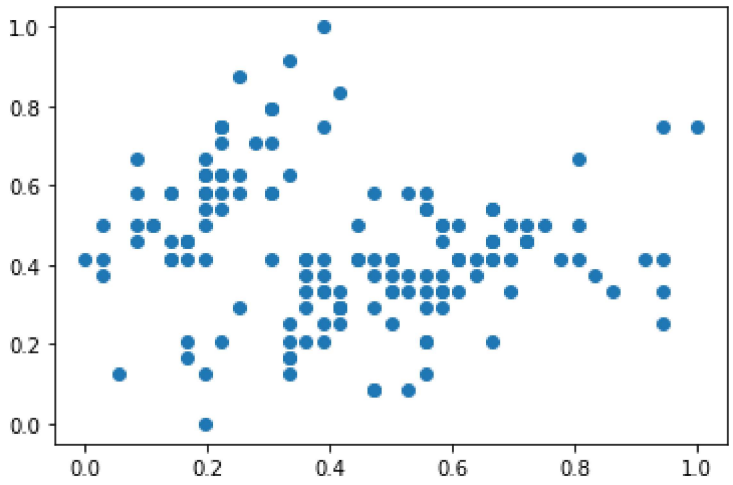
In [39]:
```
df.head()
```

Out[39]:

	s_length	s_width	p_length	p_width	target	Cluster
0	0.222222	0.625000	1.4	0.2	0	1
1	0.166667	0.416667	1.4	0.2	0	1
2	0.111111	0.500000	1.3	0.2	0	1
3	0.083333	0.458333	1.5	0.2	0	1
4	0.194444	0.666667	1.4	0.2	0	1

In [40]:
```
plt.scatter(df['s_length'],df['s_width'])
```

Out[40]:
```
<matplotlib.collections.PathCollection at 0x1c30dccaf70>
```



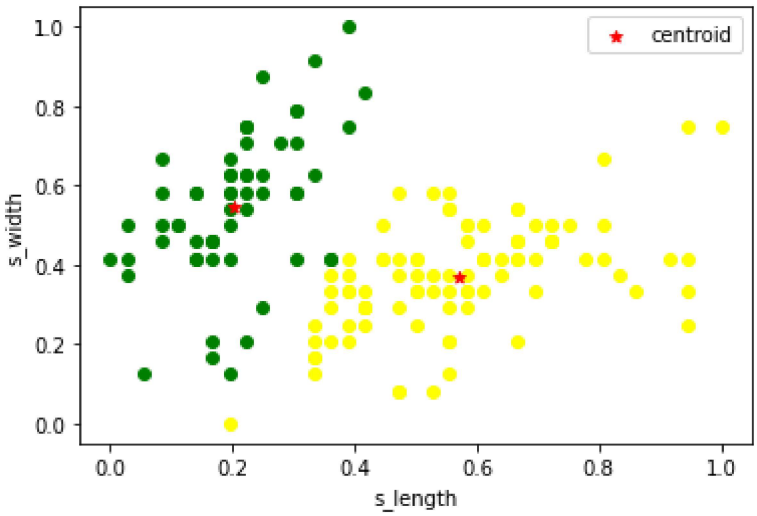
In [42]:
```
km=KMeans(n_clusters=2)
y_pred=km.fit_predict(df[['s_length','s_width']])
y_pred
```

Out[42]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

In [43]:
```
df['Cluster1']=y_pred
```

In [45]:
```
df1=df[df['Cluster1']==0]
df2=df[df['Cluster1']==1]
df3=df[df['Cluster1']==2]
plt.scatter(df1['s_length'],df1['s_width'],color='green')
plt.scatter(df2['s_length'],df2['s_width'],color="yellow")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='red',marker=
plt.xlabel("s_length")
plt.ylabel("s_width")
plt.legend()
```

Out[45]: <matplotlib.legend.Legend at 0x1c30de2f0a0>



In [ ]: