60. Silhouette Score (Practical)

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
dataset = pd.read_csv(r'Data/iris_raw.csv')
dataset.head(3)
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2

54.1 Making Clusters of Data

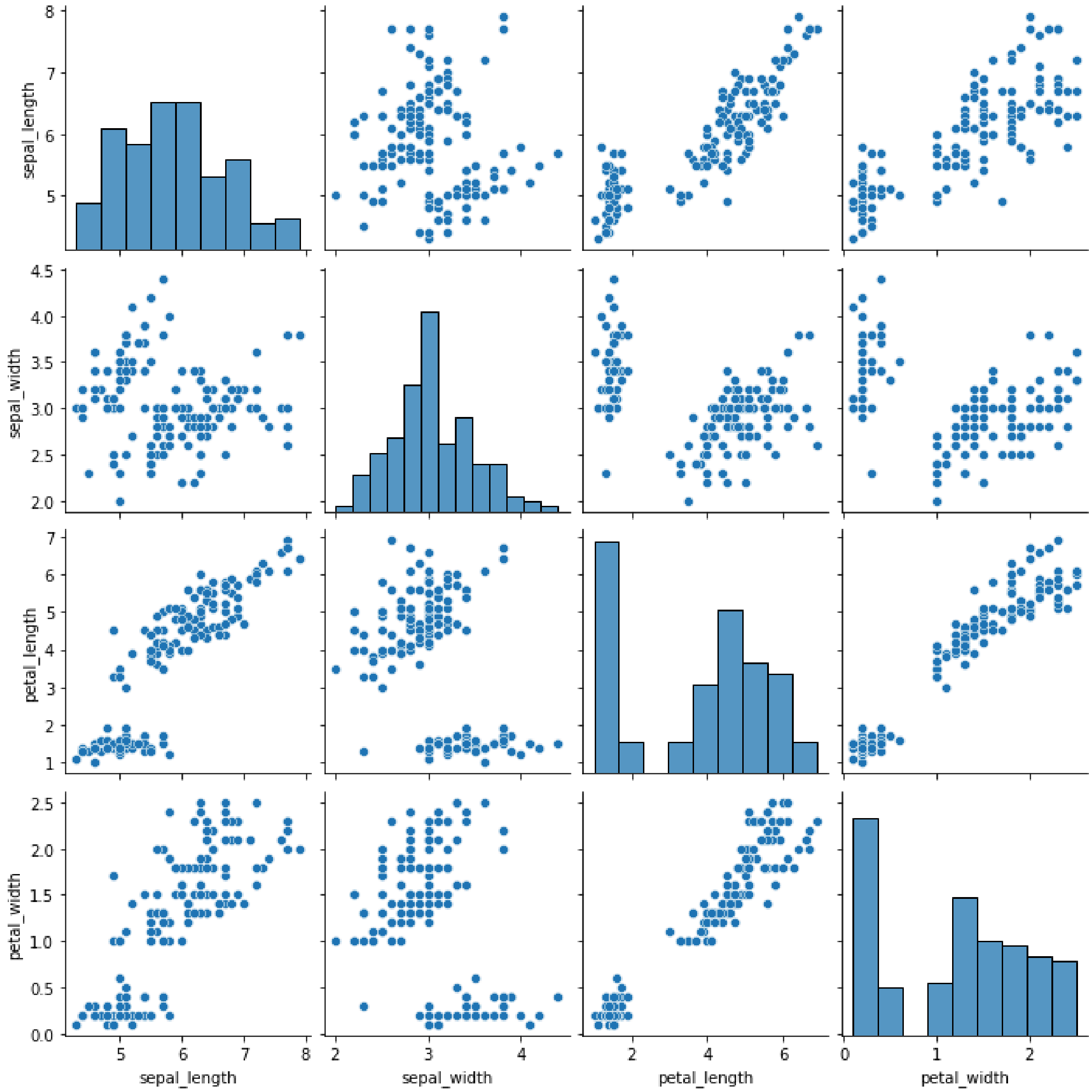
- Use K-mean clustering when **your data is linearly separable**

Check the data if it is linearly separable

In [3]:

```
sns.pairplot(data=dataset)
plt.show()
```

```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axis
grid.py:123: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



- In supervised learning, the data is split into training and testing data
- In unsupervised learning, data is not split into training and testing data b/c the data is unlabelled

54.1.1 Find Number of clusters

In [4]:

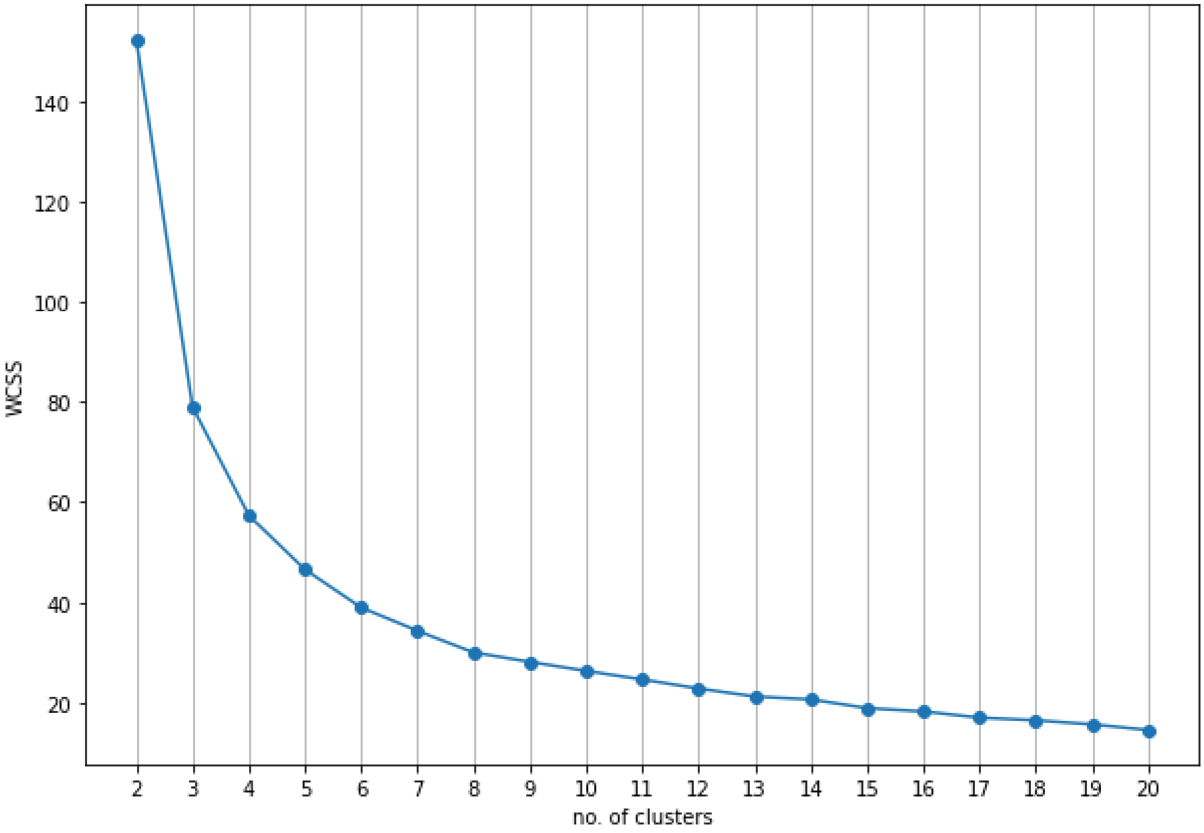
```
from sklearn.cluster import KMeans
```

In [5]:

```
# Use a loop to find best number of clusters from 2 to 20
wcss = []

for i in range(2,21):
    km = KMeans(n_clusters=i, init='k-means++')
    km.fit(dataset)
    wcss.append(km.inertia_) # it assings value of wcss {Elbow graph}
```

```
In [6]: plt.figure(figsize=(10,7))
        plt.plot([i for i in range(2,21)], wcss, marker='o')
        plt.xlabel('no. of clusters')
        plt.xticks([i for i in range(2,21)])
        plt.ylabel('WCSS')
        plt.grid(axis='x')
        plt.show()
```



Elbow point = 3

It means that will have 3 number of clusters

```
In [7]: kmn = KMeans(n_clusters=3)
        kmn.fit_predict(dataset)
```

```
Out[7]: 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, 1, 2, 2, 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, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
               2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2, 0, 0, 0, 0, 2, 0, 0, 0,
               0, 0, 0, 2, 2, 0, 0, 0, 0, 2, 0, 2, 0, 2, 0, 0, 2, 2, 0, 0, 0, 0,
               0, 2, 0, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 2])
```

```
In [8]: dataset['Predict'] = kmn.fit_predict(dataset)
```

```
In [9]: dataset
```

Out[9]:

	sepal_length	sepal_width	petal_length	petal_width	Predict
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	1
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	1

150 rows × 5 columns

54.2 Apply Silhouette Score to validate above results

In [17]:

```
from sklearn.metrics import silhouette_score
```

In [18]:

```
kmn.labels_
```

Out[18]:

```
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, 0, 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, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2,
       2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 2, 1, 2, 1, 2, 2, 1, 1, 2, 2, 2, 2,
       2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 1])
```

In [19]:

```
silhouette_score(dataset, labels=kmn.labels_)
```

Out[19]:

```
0.6126634972047179
```

We are not sure about the results. So will apply loop to determine which silhouette_score is best and determine what is acutal number of clusters.

In [29]:

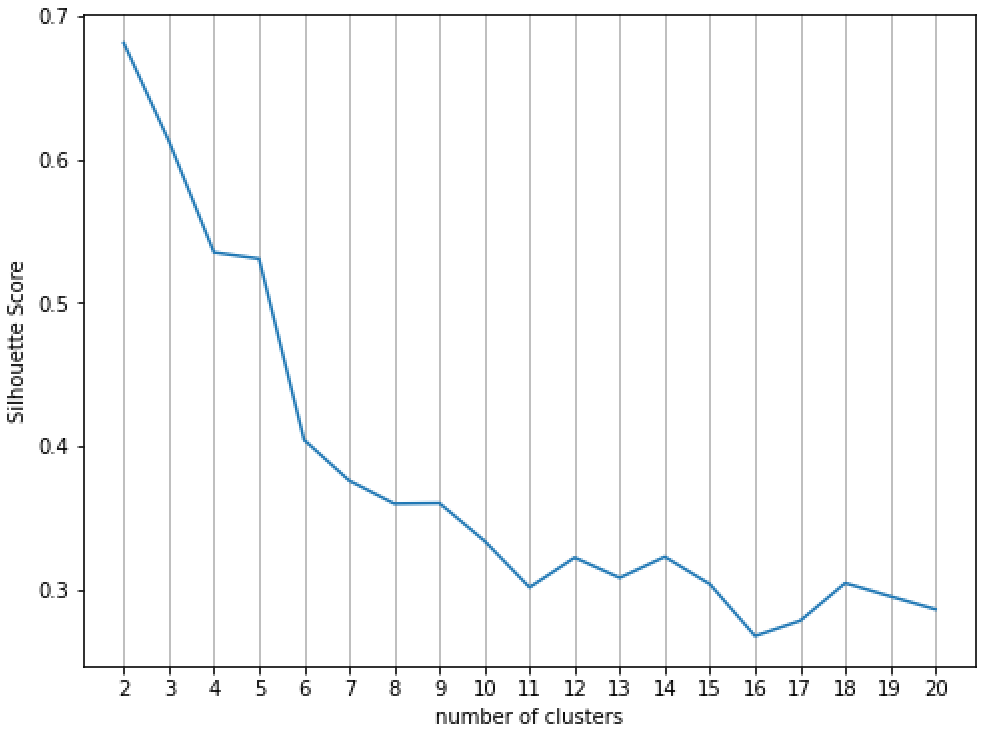
```
ss = []
n_clusters= [j for j in range(2,21)]

for i in range(2,21):
    kmn1 = KMeans(n_clusters=i)
    kmn1.fit(dataset)
    ss.append(silhouette_score(dataset, labels=kmn1.labels_))
```

- We are going to make graph b/w ss vs #clusters

In [39]:

```
plt.figure(figsize=(8,6))
plt.plot(n_clusters, ss)
plt.xlabel('number of clusters')
plt.ylabel('Silhouette Score')
plt.xticks(n_clusters)
plt.grid(axis='x')
plt.show()
```



Best Silhouette Score = 2

In [ ]: