Task-02

=> Predicting optimum number of clusters

In [17]:

```
# Importing Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [18]:

```
dataset=pd.read_csv("Iris.csv")
```

In [19]:

```
dataset.head()
```

Out[19]:

	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

Data Preprocessing

In [20]:

```
dataset.info()
```

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

In [21]:

```
dataset.isnull().sum()
```

Out[21]:

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [22]:

```
dataset["Species"].value_counts()
```

Out[22]:

```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

Converting String data to Integer

In [23]:

```
from sklearn.preprocessing import LabelEncoder
l1=LabelEncoder()
dataset["Species"]=l1.fit_transform(dataset["Species"])
```

In [24]:

```
dataset.head()
```

Out[24]:

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

In [32]:

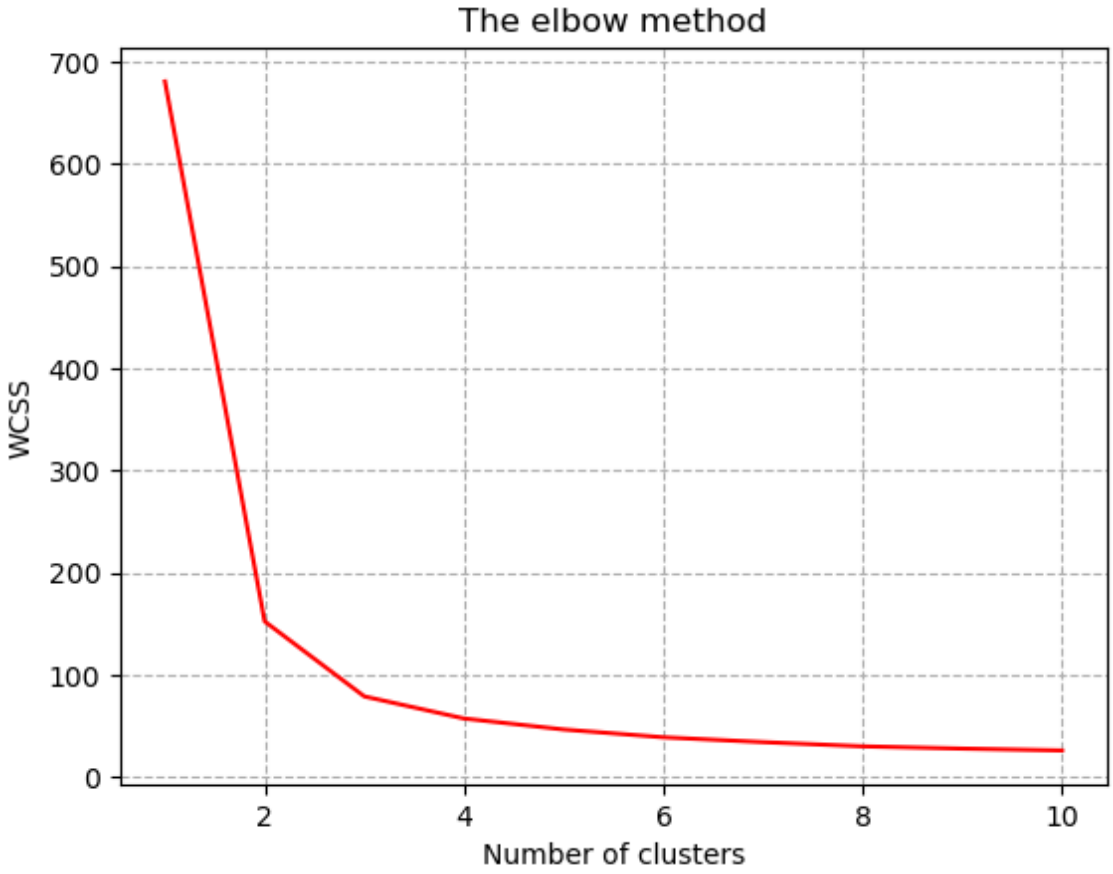
```
x = dataset.iloc[:, [1, 2, 3, 4]].values
```

Elbow Method

In [62]:

```
# Solving the problem by K-means Clustering
# By using elbow method finding optimal number of clusters


from sklearn.cluster import KMeans
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++',random_state = 0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
    print(wcss)


plt.plot(range(1, 11), wcss,"r")
plt.title('The elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.grid(ls="--")
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:103
6: UserWarning: KMeans is known to have a memory leak on Windows with MK
L, when there are less chunks than available threads. You can avoid it by
setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(

[680.8244]
[680.8244, 152.36870647733906]
[680.8244, 152.36870647733906, 78.94084142614601]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016, 38.930963049671746]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016, 38.930963049671746, 34.19068792479663]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016, 38.930963049671746, 34.19068792479663, 30.06387443273313]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016, 38.930963049671746, 34.19068792479663, 30.06387443273313,
27.842356060606065]
[680.8244, 152.36870647733906, 78.94084142614601, 57.31787321428571, 46.5
6163015873016, 38.930963049671746, 34.19068792479663, 30.06387443273313,
27.842356060606065, 26.048202248044355]
```



hat point is

Lets work on Model Training

In [34]:

```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', random_state = 0)
y_kmeans = kmeans.fit_predict(x)
```

In [35]:

```
dataset["Output"]=y_kmeans
dataset
```

Out[35]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	Output
0	1	5.1	3.5	1.4	0.2	0	1
1	2	4.9	3.0	1.4	0.2	0	1
2	3	4.7	3.2	1.3	0.2	0	1
3	4	4.6	3.1	1.5	0.2	0	1
4	5	5.0	3.6	1.4	0.2	0	1
...	...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	2	2
146	147	6.3	2.5	5.0	1.9	2	0
147	148	6.5	3.0	5.2	2.0	2	2
148	149	6.2	3.4	5.4	2.3	2	2
149	150	5.9	3.0	5.1	1.8	2	0

150 rows × 7 columns

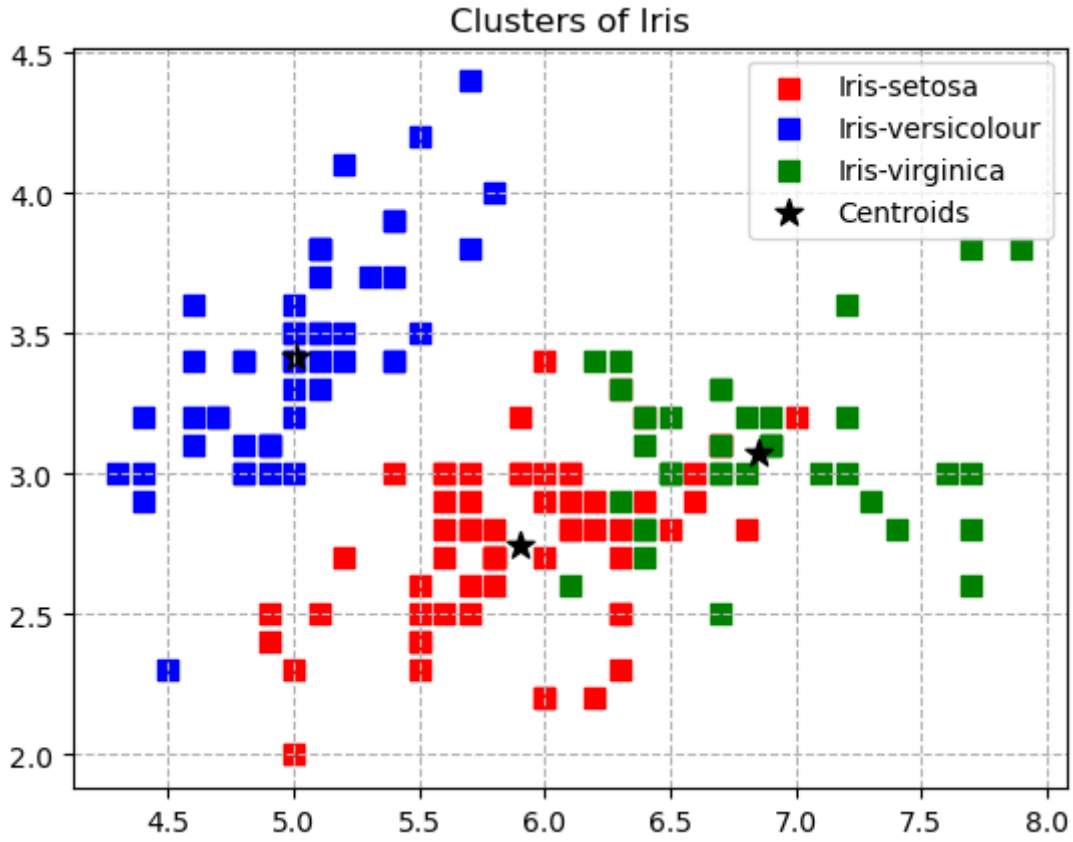
Visualization Of Clusters

In [60]:

```
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1],marker='s',s = 60, c = 'red', label
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1],marker='s',s = 60, c = 'blue', labe
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1],marker='s',s = 60, c = 'green', lab
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,1],marker='*', s =
plt.grid(ls='--')
plt.title('Clusters of Iris')
plt.legend()
```

Out[60]:

```
<matplotlib.legend.Legend at 0x26a3535d640>
```



In [ ]: