# The Sparks Foundation GRIP #JULY22

# Data Science and Business Analytics Internship ¶

# Task 2 : Prediction Using Unsupervised Machine Learning!

# By PUTTURU LIHKITHA

Problem statement

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

In [1]:

```
# Importing All Important libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
```

In [2]:

```
#Loading the dataset
iris = load_iris()
data = pd.DataFrame(iris.data,columns=iris.feature_names)
data.head()
```

Out[2]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) |
|---|---|---|---|---|
| 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 |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 |

In [3]:

```
#To know the shape of the data
data.shape
```

Out[3]:

```
(150, 4)
```

In [4]:
```
#To know the informtion of the data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
sepal length (cm)    150 non-null float64
sepal width (cm)     150 non-null float64
petal length (cm)    150 non-null float64
petal width (cm)     150 non-null float64
dtypes: float64(4)
memory usage: 4.8 KB
```

In [5]:
```
data.describe() #Describing the data
```

Out[5]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) |
|---|---|---|---|---|
| count | 150.000000 | 150.000000 | 150.000000 | 150.000000 |
| mean | 5.843333 | 3.057333 | 3.758000 | 1.199333 |
| std | 0.828066 | 0.435866 | 1.765298 | 0.762238 |
| 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 |

# Handling the null values

In [6]:
```
# Now, we handle the null values that are present in the dataset for better accu
```

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

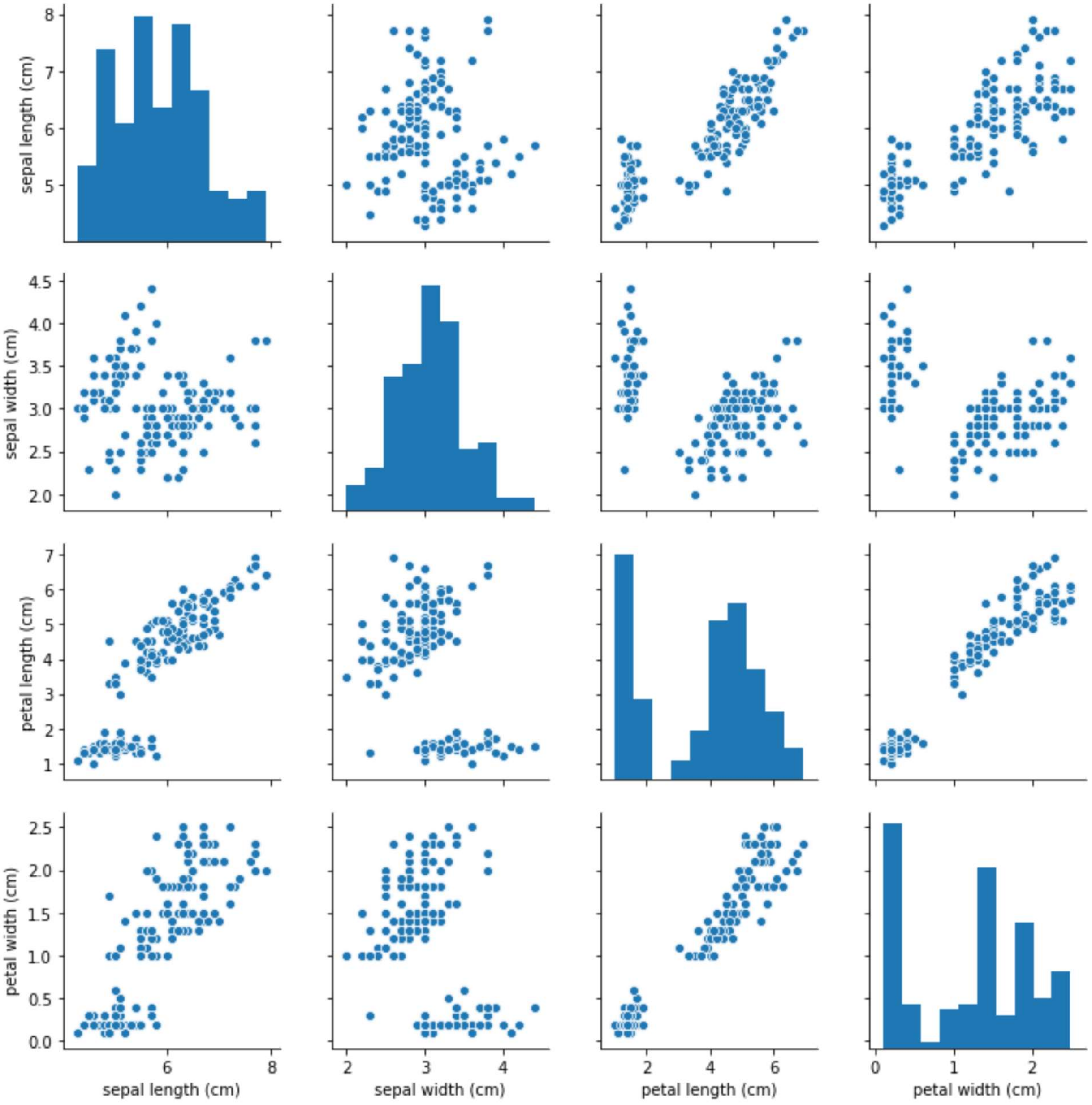
Out[7]:
```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
dtype: int64
```

# Pairplot of dataframe

In [8]:
```
#Pair plot:
# It plots a pairwise relationship in the dataset, it will create a grid of axis
```

In [9]:
```
sns.pairplot( data)
```

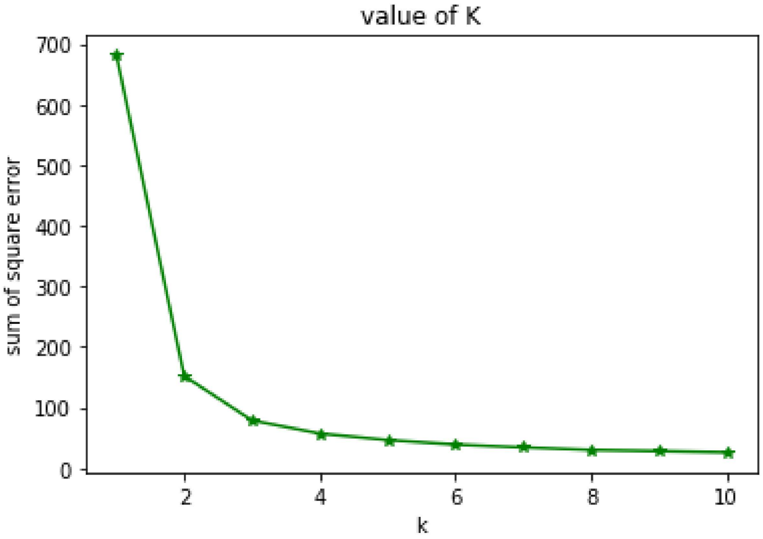
Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x1facb2e5dc8>
```



In [10]:
```
data.shape #Shape of the current data
```

Out[10]:
```
(150, 4)
```

In [11]:
```
# Here we are finding the optimaal number of clusters for k-means classification
x = data.iloc[:, [0, 1, 2, 3]].values
from sklearn.cluster import KMeans
sse = []
for i in range(1,11):
    km = KMeans(n_clusters = i , random_state = 0)
    km.fit(x)
    sse.append(km.inertia_)
plt.plot(range(1,11), sse, color = 'green' , marker = '*')
plt.title("value of K")
plt.xlabel("k")
plt.ylabel("sum of square error")
plt.show()
```



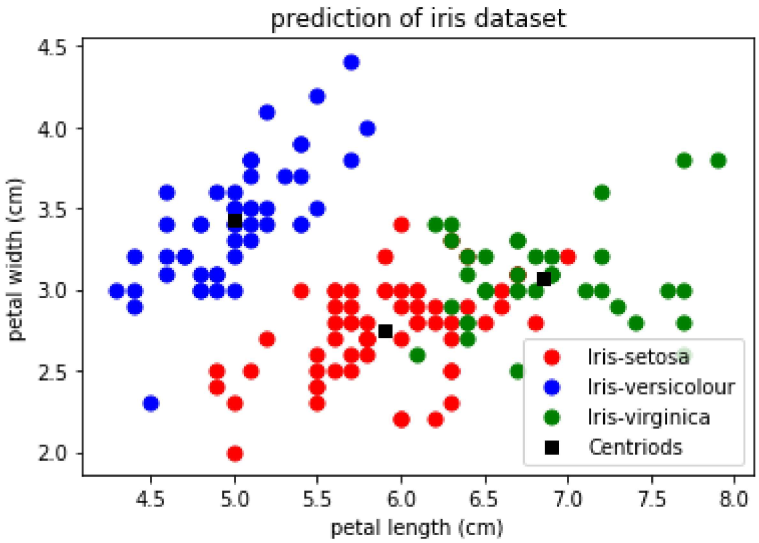
In [12]:
```
# fitting the KMeans with using the value of k = 2
model = KMeans(n_clusters = 3 , random_state = 0)
y_means = model.fit_predict(x)
y_means
```

Out[12]:
```
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, 0, 0, 2, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 2, 2, 2, 0, 2, 2, 2,
       2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
       2, 0, 2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0])
```

In [13]:
```
plt.scatter(x[y_means == 0,0],x[y_means == 0,1],c = 'red', s= 50 , label = 'Iris
plt.scatter(x[y_means == 1,0], x[y_means == 1 , 1] , c = 'blue' ,s=50, label = '
plt.scatter(x[y_means == 2,0] , x[y_means == 2,1] , c = 'green', s = 50 , label 
plt.scatter(model.cluster_centers_[:,0], model.cluster_centers_[:,1],marker = 's
plt.title("prediction of iris dataset")
plt.xlabel('petal length (cm)')
plt.ylabel('petal width (cm)')


plt.legend()
```

Out[13]:
```
<matplotlib.legend.Legend at 0x1facc586488>
```



# Here this concludes the K-Map clustering

In [ ]: