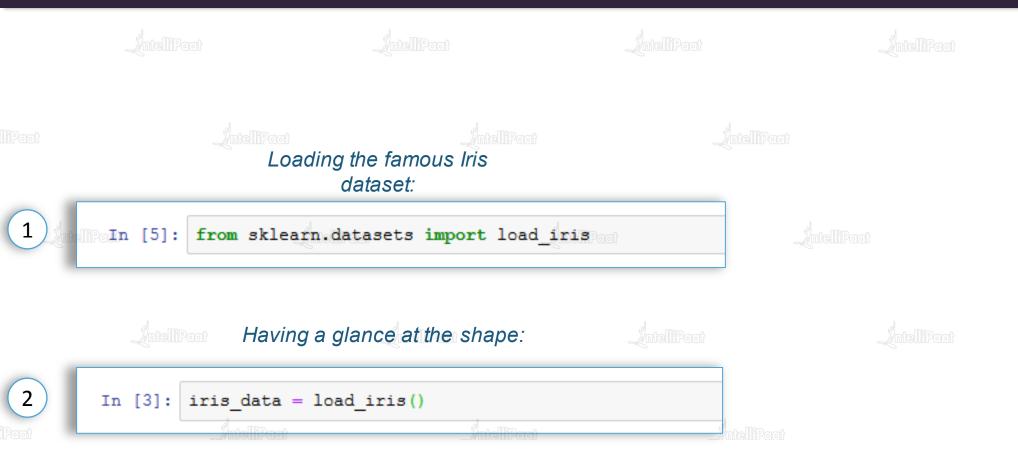# Hands-on: K-means Algorithm

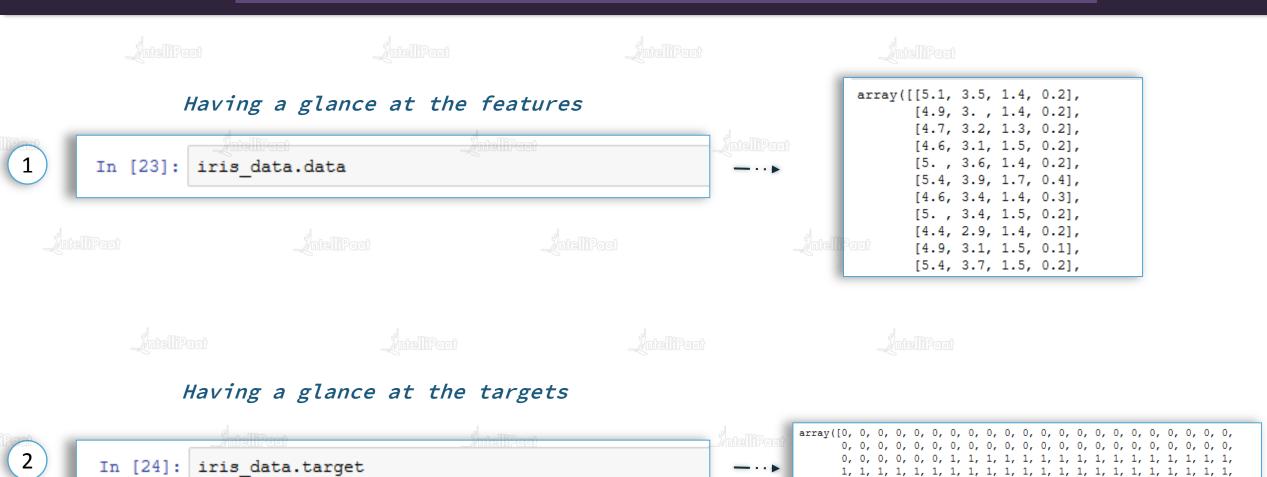# Demo-
K-means
Algorithm

- We will be using the famous Iris Dataset, collected in the 1930's by Edgar Anderson.

- In this example, we are going to train a random forest classification algorithm to predict the class in the test data

	A	B	C	D	E	F
1	class	petal_len	petal_wid	sepal_len	sepal_width	
2	Iris-virgini	5.5	1.8	6.4	3.1	
3	Iris-virgini	5.9	2.3	6.8	3.2	
4	Iris-virgini	5.4	2.3	6.2	3.4	
5	Iris-virgini	4.8	1.8	6	3	
6	Iris-virgini	5.1	2.3	6.9	3.1	
7	Iris-virgini	5.6	2.4	6.3	3.4	
8	Iris-virgini	5.2	2.3	6.7	3	
9	Iris-virgini	6.7	2	7.7	2.8	
10	Iris-virgini	5.8	2.2	6.5	3	
11	Iris-virgini	5.3	1.9	6.4	2.7	
12	Iris-virgini	5	2	5.7	2.5	
13	Iris-virgini	5.1	1.9	5.8	2.7	

Loading the famous Iris dataset:

1

```
In [5]: from sklearn.datasets import load_iris
```

Having a glance at the shape:

2

```
In [3]: iris_data = load_iris()
```

Having a glance at the features

1

```
In [23]:   iris_data.data
```

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
```

Having a glance at the targets

2

```
In [24]:   iris_data.target
```

```
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, 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, 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])
```

Building the model:

1

```
In [12]: from sklearn.cluster import KMeans
```

Set up the number of clusters:

2

```
In [25]: # set the number of clusters
         kmeans = KMeans(n_clusters = 3)
```
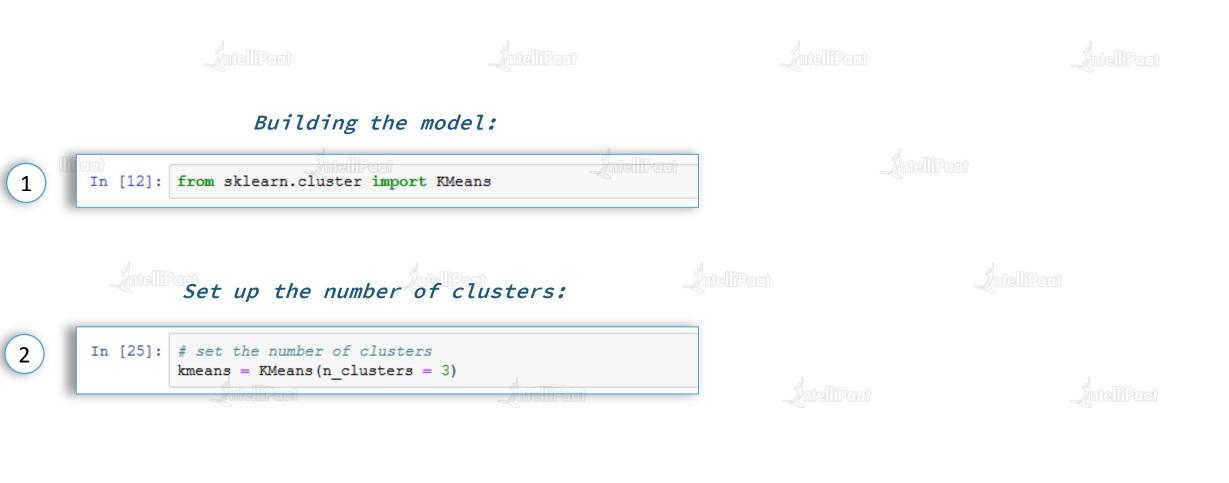
www.intellipaat.c

K-means Algorithm

Fit the features into the model:

1

```
In [26]:  kmodel = kmeans.fit(iris_data.data)
```

Predict and label the data:

2

```
In [32]:  kmodel1 = kmeans.predict(iris_data.data)
          kmodel1
```

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

Having a look at the cluster centers:

3

```
In [20]:  kmodel.cluster_centers_
```

```
array([[5.006     , 3.428     , 1.462     , 0.246     ],
       [5.9016129 , 2.7483871 , 4.39354839, 1.43387097],
       [6.85      , 3.07368421, 5.74210526, 2.07105263]])
```

Evaluating the Algorithm:

```
In [22]:  import pandas as pd
          pd .crosstab(iris_data.target, kmodel.labels_)
```

col_0	0	1	2
row_0			
0	50	0	0
1	0	48	2
2	0	14	36

- 0's -> All 50 of them labeled as 0,

- 1's -> 48 of them labeled correctly, 2 of them labeled incorrectly

- 2's -> 36 of them labeled correctly, 14 of them labeled incorrectly