```
import sklearn
# Import scikit learn
from sklearn import datasets
# Load data
iris= datasets.load_iris()
# Print shape of data to confirm data is loaded
print(iris.data.shape)
##################################################
from sklearn import svm
from sklearn import datasets
# Load dataset
iris = datasets.load_iris()
clf = svm.LinearSVC()
# learn from the data
clf.fit(iris.data, iris.target)
# predict for unseen data
clf.predict([[ 5.0,  3.6,  1.3,  0.25]])
# Parameters of model can be changed by using the attributes ending with an underscore
print(clf.coef_ )
#import the model
##################################################
from sklearn import linear_model
reg = linear_model.LinearRegression()
# use it to fit a data
reg.fit ([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
# Let's look into the fitted data
print(reg.coef_)
# Load dataset
iris = datasets.load_iris()
# Create and fit a nearest-neighbor classifier
##################################################
```

```
from sklearn import neighbors
knn = neighbors.KNeighborsClassifier()
knn.fit(iris.data, iris.target)
# Predict and print the result
result=knn.predict([[0.1, 0.2, 0.3, 0.4]])
print(result)
from sklearn import cluster, datasets
# load data
iris = datasets.load_iris()
# create clusters for k=3
k=3
k_means = cluster.KMeans(k)
# fit data
k_means.fit(iris.data)
# print results
print( k_means.labels_[::10])
print( iris.target[::10])
```

Output

```
(150, 4)
[[ 0.18344977  0.4549369  -0.81495365 -0.43357339]
 [ 0.05524625 -0.9008946   0.40923385 -0.9606271 ]
 [-0.85048831 -0.98657611  1.38090768  1.86511359]]
[0.5 0.5]
[0]
[0 0 0 0 0 1 1 1 1 1 2 2 2 2 2]
[0 0 0 0 0 1 1 1 1 1 2 2 2 2 2]
```