```
from sklearn.linear_model import Perceptron
from sklearn.metrics import accuracy_score
import numpy as np
```

```
data = [[1.81, 0.80, 0.44],
[1.77, 0.70, 0.43],
[1.60, 0.60, 0.38],
[1.54, 0.54, 0.37],
[1.66, 0.65, 0.40],
[1.90, 0.90, 0.47],
[1.75, 0.64, 0.39],
[1.77, 0.70, 0.40],
[1.59, 0.55, 0.37],
[1.71, 0.75, 0.42],
[1.81, 0.85, 0.43]]
```

```
results = ['male', 'male',
'female', 'female',
'male', 'male',
'female', 'female','female',
'male', 'male']
```

```
model = Perceptron (alpha=0.0001, class_weight=None, eta0=1.0, fit_intercept=True, max_iter=1000, n_jobs=1, penalty=None, random_state=0,
```

```
model.fit(data,results)
```

```
Perceptron
Perceptron(n_jobs=1)
```

```
predicted_results = model.predict(data)
acc_per = accuracy_score(results, predicted_results) * 100
print('Accuracy for perceptron: {} %'.format(acc_per))
```

```
Accuracy for perceptron: 54.54545454545454 %
```

```
prediction = model.predict([[1.62, 0.49, 0.38]])
print(prediction)
```

```
['male']
```

```
import numpy as np
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.linear_model import Perceptron
from sklearn.neighbors import NearestNeighbors
```

```
methods = ['Decision Trees', 'SVM', 'Perceptron', 'K nearest neighbour']
```

```
X = [[181, 88, 44], [177, 70, 43], [160, 60, 38], [154, 54, 37], [166, 65, 40], [190, 90, 47], [175, 64, 39], [177, 78, 40], [159, 55, 37
Y = ['male', 'male', 'female', 'female', 'male', 'male', 'female', 'female', 'female', 'male', 'male']
```

```
clf_tree = DecisionTreeClassifier()
clf_svm = SVC()
clf_percept = Perceptron()
clf_KNN = NearestNeighbors()
```

```
clf_tree = clf_tree.fit(X,Y)
clf_svm = clf_svm.fit(X,Y)
clf_percept = clf_percept.fit(X,Y)
c1f_KNN = clf_KNN.fit(X,Y)
```

```
clf_tree_prediction = clf_tree.predict(X)
acc_tree = accuracy_score(Y, clf_tree_prediction)*100
print ("Accuracy using Decision Trees:"), acc_tree, "%"
```

```
Accuracy using Decision Trees:
(None, 100.0, '%')
```

```
clf_svm_prediction = clf_svm.predict(X)
acc_svm = accuracy_score (Y, clf_svm_prediction)*100
```

```
print ("Labels for training set using SVM:'"),acc_svm, "%"
```

```
Labels for training set using SVM:'
(None, 54.54545454545454, '%')
```

```
clf_percept_prediction = clf_percept.predict(X)
acc_per = accuracy_score (Y, clf_percept_prediction)*100
print ("Labels for training set using Perceptron:"), acc_per, "%"
```

```
Labels for training set using Perceptron:
(None, 45.45454545454545, '%')
```

```
distances, indices = clf_KNN.kneighbors (X)
new_label = indices[:,0]
clf_KNN_prediction = [Y[i][:] for i in new_label ]
acc_knn = accuracy_score(Y, clf_KNN_prediction)*100
print ("Labels for training set using K-nearst neighbour:"), acc_knn, "%"
```

```
Labels for training set using K-nearst neighbour:
(None, 100.0, '%')
```

```
acc_all = [acc_tree,acc_svm,acc_per,acc_knn]
score_bestmethod = np.max(acc_all)
best_method = np.argmax(acc_all)
```

```
print (methods[best_method], "is the best method with accuracy of"), score_bestmethod, "%"
```

```
Decision Trees is the best method with accuracy of
(None, 100.0, '%')
```