In [1]:

```
%matplotlib notebook
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

np.set_printoptions(precision=2)

from sklearn.datasets import make_regression
X_R1, y_R1=make_regression(n_samples=100,n_features=4,
                           n_informative=1,bias=150.0,
                           noise=30,random_state=0)
```

In [2]:

```
#for classification
fruits=pd.read_table('fruit_data_with_colors.txt')

X_fruits=fruits[['height','width','mass','color_score']]
y_fruit=fruits['fruit_label']

X_fruits_2d=fruits[['height','width','mass','color_score']]
y_fruits_2d=fruits['fruit_label']
```

In [3]:

```
print(X_fruits_2d.shape)
```

```
(59, 4)
```

In [4]:

```
print(y_fruits_2d.shape)
```

```
(59,)
```

In [6]:

```
clfList=[]
clfTestList=[]
clfTrainingList=[]
```

In [9]:

```
#Logistic Regression
```

In [7]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.multiclass import OneVsOneClassifier


X_train,X_test,y_train,y_test=train_test_split(X_fruits_2d,
                                               y_fruits_2d,
                                               random_state=0)
clf=OneVsOneClassifier(LogisticRegression()).fit(X_train, y_train)


print('Accuracy of Logistic classifier on training set:{:.2f}'
     .format(clf.score(X_train,y_train)))
print('Accuracy of Logistic classifier on test set:{:.2f}'
     .format(clf.score(X_test,y_test)))
clfList.append(clf)
clfTestList.append(clf.score(X_test,y_test))
clfTrainingList.append(clf.score(X_train,y_train))
```

```
Accuracy of Logistic classifier on training set:0.82
Accuracy of Logistic classifier on test set:0.53
```

In [ ]:

```
##Support Vector Machine
```

In [8]:

```
from sklearn.svm import SVC

X_train,X_test,y_train,y_test=train_test_split(X_fruits_2d,
                                               y_fruits_2d,
                                               random_state=0)

clf=OneVsOneClassifier(SVC(kernel='linear')).fit(X_train,y_train)
#
print('Accuracy of SVC classifier on training set:{:.2f}'
     .format(clf.score(X_train,y_train)))
print('Accuracy of SVC classifier on test set:{:.2f}'
     .format(clf.score(X_test,y_test)))
clfList.append(clf)
clfTestList.append(clf.score(X_test,y_test))
clfTrainingList.append(clf.score(X_train,y_train))
```

```
Accuracy of SVC classifier on training set:0.82
Accuracy of SVC classifier on test set:0.60
```

In [12]:

```
## Decision Tree
```

In [9]:

```
from sklearn.tree import DecisionTreeClassifier

X_train,X_test,y_train,y_test=train_test_split(X_fruits_2d,
                                               y_fruits_2d,
                                               random_state=0)
clf=OneVsOneClassifier(DecisionTreeClassifier()).fit(X_train,y_train)
#.
print('Accuracy of DecisionTree classifier on training set:{:.2f}'
     .format(clf.score(X_train,y_train)))
print('Accuracy of DecisionTree classifier on test set:{:.2f}'
     .format(clf.score(X_test,y_test)))
clfList.append(clf)
clfTestList.append(clf.score(X_test,y_test))
clfTrainingList.append(clf.score(X_train,y_train))
```

```
Accuracy of DecisionTree classifier on training set:1.00
Accuracy of DecisionTree classifier on test set:0.80
```

In [10]:

```
##Random Forest
```

In [11]:

```
from sklearn.ensemble import RandomForestClassifier

X_train,X_test,y_train,y_test=train_test_split(X_fruits_2d,
                                               y_fruits_2d,
                                               random_state=0)
clf=OneVsOneClassifier(RandomForestClassifier(n_estimators=12,random_state=0)).fit(X_train,y_tra
in)
#.
print('Accuracy of RandomForest classifier on training set:{:.2f}'
     .format(clf.score(X_train,y_train)))
print('Accuracy of RandomForest classifier on test set:{:.2f}'
     .format(clf.score(X_test,y_test)))
clfList.append(clf)
clfTestList.append(clf.score(X_test,y_test))
clfTrainingList.append(clf.score(X_train,y_train))
```

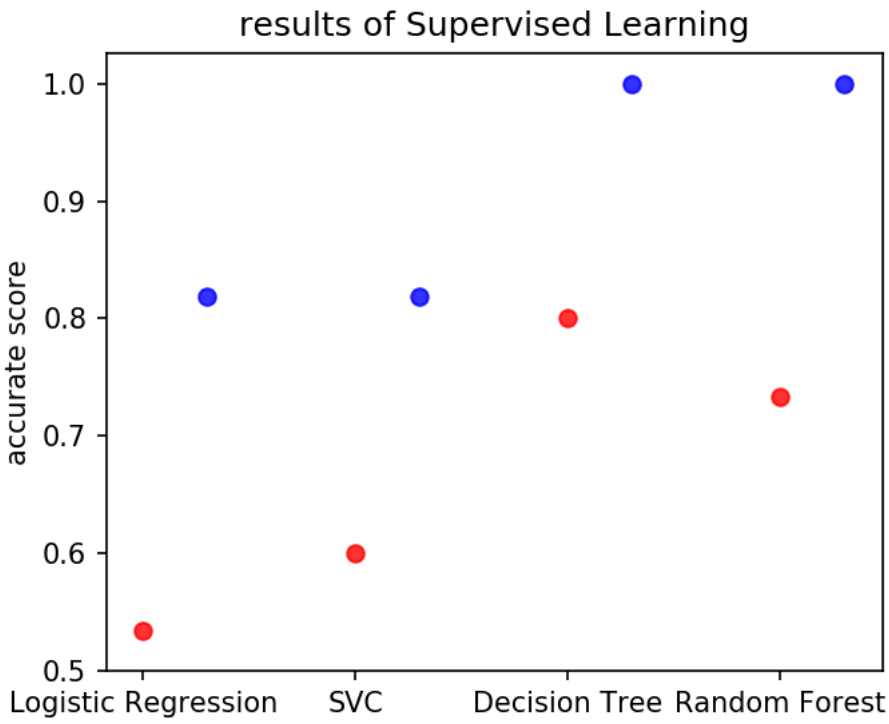
```
Accuracy of RandomForest classifier on training set:1.00
Accuracy of RandomForest classifier on test set:0.73
```

In [242]:

```
#1. 어떤 모델에서 테스트셋 구분에 가장 좋은 성능을 보이는지 간단한 그래프를 통해 구하세요.
```

In [12]:

```
plt.figure(figsize=(5,4))
x_axis=range(len(clfTestList))
scatters=plt.scatter(x_axis,clfTestList,marker='o',alpha=0.8,color='red')
```



In [13]:

```
names=['Logistic Regression','SVC','Decision Tree','Random Forest']
pos=np.arange(len(names))
plt.xticks(pos,names)
plt.ylabel('accurate score')
plt.title('results of Supervised Learning')
```

Out[13]:

```
Text(0.5, 1, 'results of Supervised Learning')
```

In [14]:

```
new_xvals=[]
for i in x_axis:
    new_xvals.append(i+0.3)
scatters2=plt.scatter(new_xvals,clfTrainingList,marker='o',alpha=0.8,color='blue')
```

In [ ]:

```
#1. Decistion Tree에서 가장 좋은 성능을 보입니다.
```

In [ ]:

```
## 2. 그리고 그 때, [무게가 120, 너비가 6, 높이가 8, color_score가 0.7]인 과일은 무엇인지

#그 모델을 가지고 추정하세요.
```

In [19]:

```
show_fruit_name=dict(zip(fruits.fruit_label.unique(),fruits.fruit_name.unique()))
```

In [21]:

```
#X_fruits=fruits[['height','width','mass','color_score']]
#y_fruit=fruits['fruit_label']
prediction=clf.predict([[8,6,120,0.7]])
show_fruit_name[prediction[0]]
```

Out[21]:

```
'lemon'
```

In [ ]:

```
# 2.lemon입니다.
```

In [ ]:

```

```