In [48]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
import numpy as np
from sklearn import tree
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
```

In [49]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(iris.data,iris.target,test_size=0.3)
```

In [50]:
```
lr = LogisticRegression() #logisticregression
lr.fit(x_train, y_train)
lr.score(x_test, y_test)
```

Out[50]: 1.0

In [51]:
```
svm =SVC() #svm
svm.fit(x_train, y_train)
svm.score(x_test, y_test)
```

Out[51]: 0.9777777777777777

In [52]:
```
rf = RandomForestClassifier(n_estimators=40) #randomforest
rf.fit(x_train, y_train)
rf.score(x_test, y_test)
```

Out[52]: 1.0

In [53]:
```
tree = DecisionTreeClassifier()
tree.fit(x_train, y_train)
tree.score(x_test, y_test)
```

Out[53]: 0.9555555555555556

In [54]:
```
from sklearn.model_selection import cross_val_score
```

In [55]:
```
LR=cross_val_score(LogisticRegression(), iris.data, iris.target,cv=3)
LR
```

```
C:\Users\Sai Sushma Iska\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[55]: array([0.98, 0.96, 0.98])

In [56]:
```
svm = cross_val_score(SVC(), iris.data, iris.target,cv=3)
svm
```

Out[56]: array([0.96, 0.98, 0.94])

In [57]:
```
RF = cross_val_score(RandomForestClassifier(n_estimators=40),iris.data, iris.target,cv=3)
RF
```

Out[57]: array([0.98, 0.92, 0.92])

In [58]:
```
tree = cross_val_score(DecisionTreeClassifier(), iris.data, iris.target,cv=3)
tree
```

Out[58]: array([0.98, 0.94, 0.98])

In [59]:
```
np.average(LR)
```

Out[59]: 0.9733333333333333

In [60]:
```
np.average(svm)
```

Out[60]: 0.96

In [61]:
```
np.average(RF)
```

Out[61]: 0.94

In [47]:
```
np.average(tree)
```

Out[47]: 0.96

In [ ]: