scikit-learn

Evaluate several classifiers on the Wine dataset using scikit-learn

Ratio 60-40

Svm

In [1]:

```python
from sklearn import svm
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# Support Vector Machine (SVM)
clf_svm = svm.SVC()
clf_svm.fit(x_train, y_train)

# Accuracy Scores
svm_acc_test = clf_svm.score(x_test, y_test)
svm_acc_train = clf_svm.score(x_train, y_train)

# Cross-Validation
cv_svm = cross_val_score(clf_svm, x, y, cv=5)

# Printing Results
print("SVM Accuracy on Test Set:", svm_acc_test)
print("SVM Accuracy on Training Set:", svm_acc_train)
print("SVM Cross-Validation Scores:", cv_svm)
print("SVM Cross-Validation Scores percentage :", cv_svm * 100)
print("SVM Cross-Validation Mean Score:", cv_svm.mean())
print("SVM Cross-Validation Standard Deviation:", cv_svm.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
SVM Accuracy on Test Set: 0.011299435028248588
SVM Accuracy on Training Set: 0.16981132075471697
SVM Cross-Validation Scores: [0.01123596 0.01123596 0.02272727 0.02272727 0.02272727]
SVM Cross-Validation Scores percentage : [1.12359551 1.12359551 2.27272727 2.27272727 2.27272727]
SVM Cross-Validation Mean Score: 0.018130745658835545
SVM Cross-Validation Standard Deviation: 0.005629572953281053
```

Decision Tree Classifier

In [1]:
```python
from sklearn.datasets import  load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier

# Loading the dataset
x, y =  load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)

# Accuracy Scores
dt_acc_test = dt.score(x_test, y_test)
dt_acc_train = dt.score(x_train, y_train)

# Cross-Validation
cv_dt = cross_val_score(dt, x, y, cv=5)

# Printing Results
print("Decision Tree Accuracy on Test Set:", dt_acc_test)
print("Decision Tree Accuracy on Training Set:", dt_acc_train)
print("Decision Tree Cross-Validation Scores:", cv_dt)
print("Decision Tree Cross-Validation Mean Score:", cv_dt.mean())
print("Decision Tree Cross-Validation Standard Deviation:", cv_dt.std())
```

```
Decision Tree Accuracy on Test Set: 0.011299435028248588
Decision Tree Accuracy on Training Set: 1.0
Decision Tree Cross-Validation Scores: [0.         0.01123596 0.         0.01136364 0.03409091]
Decision Tree Cross-Validation Mean Score: 0.011338100102145046
Decision Tree Cross-Validation Standard Deviation: 0.012448344712794315
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

Random Forest Classifier

In [2]:
```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import  load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y =  load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# Random Forest
rf = RandomForestClassifier()
rf.fit(x_train, y_train)

# Accuracy Scores
rf_acc_test = rf.score(x_test, y_test)
rf_acc_train = rf.score(x_train, y_train)

# Cross-Validation
cv_rf = cross_val_score(rf, x, y, cv=5)

# Printing Results
print("Random Forest Accuracy on Test Set:", rf_acc_test)
print("Random Forest Accuracy on Training Set:", rf_acc_train)
print("Random Forest Cross-Validation Scores:", cv_rf)
print("Random Forest Cross-Validation Mean Score:", cv_rf.mean())
print("Random Forest Cross-Validation Standard Deviation:", cv_rf.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
Random Forest Accuracy on Test Set: 0.011299435028248588
Random Forest Accuracy on Training Set: 1.0
Random Forest Cross-Validation Scores: [0.01123596 0.01123596 0.02272727 0.         0.        ]
Random Forest Cross-Validation Mean Score: 0.009039836567926455
Random Forest Cross-Validation Standard Deviation: 0.008490336528014853
```

Logistic Regression

In [3]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import  load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y =  load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# Logistic Regression
lr = LogisticRegression()
lr.fit(x_train, y_train)

# Accuracy Scores
lr_acc_test = lr.score(x_test, y_test)
lr_acc_train = lr.score(x_train, y_train)

# Cross-Validation
cv_lr = cross_val_score(lr, x, y, cv=5)

# Printing Results
print("Logistic Regression Accuracy on Test Set:", lr_acc_test)
print("Logistic Regression Accuracy on Training Set:", lr_acc_train)
print("Logistic Regression Cross-Validation Scores:", cv_lr)
print("Logistic Regression Cross-Validation Mean Score:", cv_lr.mean())
print("Logistic Regression Cross-Validation Standard Deviation:", cv_lr.std())
```

```
Logistic Regression Accuracy on Test Set: 0.005649717514124294
Logistic Regression Accuracy on Training Set: 0.026415094339622643
Logistic Regression Cross-Validation Scores: [0.01123596 0.01123596 0.01136364 0.01136364 0.01136364]
Logistic Regression Cross-Validation Mean Score: 0.01131256384065373
Logistic Regression Cross-Validation Standard Deviation: 6.255081059201201e-05
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

GaussianNB

In [5]:
```python
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import  load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y =  load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# Naive Bayes
nb = GaussianNB()
nb.fit(x_train, y_train)

# Accuracy Scores
nb_acc_test = nb.score(x_test, y_test)
nb_acc_train = nb.score(x_train, y_train)

# Cross-Validation
cv_nb = cross_val_score(nb, x, y, cv=5)

# Printing Results
print("Naive Bayes Accuracy on Test Set:", nb_acc_test)
print("Naive Bayes Accuracy on Training Set:", nb_acc_train)
print("Naive Bayes Cross-Validation Scores:", cv_nb)
print("Naive Bayes Cross-Validation Mean Score:", cv_nb.mean())
print("Naive Bayes Cross-Validation Standard Deviation:", cv_nb.std())
```

```
Naive Bayes Accuracy on Test Set: 0.0
Naive Bayes Accuracy on Training Set: 0.7509433962264151
Naive Bayes Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
Naive Bayes Cross-Validation Mean Score: 0.004545454545454545
Naive Bayes Cross-Validation Standard Deviation: 0.005567022142689042
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

K Neighbors Classifier

In [6]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# K-Nearest Neighbors (KNN)
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)

# Accuracy Scores
knn_acc_test = knn.score(x_test, y_test)
knn_acc_train = knn.score(x_train, y_train)

# Cross-Validation
cv_knn = cross_val_score(knn, x, y, cv=5)

# Printing Results
print("K-Nearest Neighbors Accuracy on Test Set:", knn_acc_test)
print("K-Nearest Neighbors Accuracy on Training Set:", knn_acc_train)
print("K-Nearest Neighbors Cross-Validation Scores:", cv_knn)
print("K-Nearest Neighbors Cross-Validation Mean Score:", cv_knn.mean())
print("K-Nearest Neighbors Cross-Validation Standard Deviation:", cv_knn.std())
```

```
K-Nearest Neighbors Accuracy on Test Set: 0.0
K-Nearest Neighbors Accuracy on Training Set: 0.16981132075471697
K-Nearest Neighbors Cross-Validation Scores: [0. 0. 0. 0. 0.]
K-Nearest Neighbors Cross-Validation Mean Score: 0.0
K-Nearest Neighbors Cross-Validation Standard Deviation: 0.0
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

AdaBoost Classifier

In [7]:
```python
from sklearn.ensemble import AdaBoostClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.6)

# AdaBoost
adaboost = AdaBoostClassifier()
adaboost.fit(x_train, y_train)

# Accuracy Scores
adaboost_acc_test = adaboost.score(x_test, y_test)
adaboost_acc_train = adaboost.score(x_train, y_train)

# Cross-Validation
cv_adaboost = cross_val_score(adaboost, x, y, cv=5)

# Printing Results
print("AdaBoost Accuracy on Test Set:", adaboost_acc_test)
print("AdaBoost Accuracy on Training Set:", adaboost_acc_train)
print("AdaBoost Cross-Validation Scores:", cv_adaboost)
print("AdaBoost Cross-Validation Mean Score:", cv_adaboost.mean())
print("AdaBoost Cross-Validation Standard Deviation:", cv_adaboost.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
AdaBoost Accuracy on Test Set: 0.0
AdaBoost Accuracy on Training Set: 0.04150943396226415
AdaBoost Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
AdaBoost Cross-Validation Mean Score: 0.004545454545454545
AdaBoost Cross-Validation Standard Deviation: 0.005567022142689042
```

Ratio 70-30

SVM

In [8]:
```
from sklearn.svm import SVC
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# Support Vector Machine (SVM)
clf_svm = SVC()
clf_svm.fit(x_train, y_train)

# Accuracy Scores
svm_acc_test = clf_svm.score(x_test, y_test)
svm_acc_train = clf_svm.score(x_train, y_train)

# Cross-Validation
cv_svm = cross_val_score(clf_svm, x, y, cv=5)

# Printing Results
print("SVM Accuracy on Test Set:", svm_acc_test)
print("SVM Accuracy on Training Set:", svm_acc_train)
print("SVM Cross-Validation Scores:", cv_svm)
print("SVM Cross-Validation Scores percentage :", cv_svm * 100)
print("SVM Cross-Validation Mean Score:", cv_svm.mean())
print("SVM Cross-Validation Standard Deviation:", cv_svm.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
SVM Accuracy on Test Set: 0.022556390977443608
SVM Accuracy on Training Set: 0.1779935275080906
SVM Cross-Validation Scores: [0.01123596 0.01123596 0.02272727 0.02272727 0.02272727]
SVM Cross-Validation Scores percentage : [1.12359551 1.12359551 2.27272727 2.27272727 2.27272727]
SVM Cross-Validation Mean Score: 0.018130745658835545
SVM Cross-Validation Standard Deviation: 0.005629572953281053
```

Decision Tree Classifier

In [9]:
```
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier

# Loading the dataset
x, y = load_wine(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)

# Accuracy Scores
dt_acc_test = dt.score(x_test, y_test)
dt_acc_train = dt.score(x_train, y_train)

# Cross-Validation
cv_dt = cross_val_score(dt, x, y, cv=5)

# Printing Results
print("Decision Tree Accuracy on Test Set:", dt_acc_test)
print("Decision Tree Accuracy on Training Set:", dt_acc_train)
print("Decision Tree Cross-Validation Scores:", cv_dt)
print("Decision Tree Cross-Validation Mean Score:", cv_dt.mean())
print("Decision Tree Cross-Validation Standard Deviation:", cv_dt.std())
```

```
Decision Tree Accuracy on Test Set: 0.9444444444444444
Decision Tree Accuracy on Training Set: 1.0
Decision Tree Cross-Validation Scores: [0.91666667 0.86111111 0.88888889 0.91428571 0.82857143]
Decision Tree Cross-Validation Mean Score: 0.8819047619047617
Decision Tree Cross-Validation Standard Deviation: 0.033414112249995145
```

Logistic Regression

In [4]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# Logistic Regression
lr = LogisticRegression()
lr.fit(x_train, y_train)

# Accuracy Scores
lr_acc_test = lr.score(x_test, y_test)
lr_acc_train = lr.score(x_train, y_train)

# Cross-Validation
cv_lr = cross_val_score(lr, x, y, cv=5)

# Printing Results
print("Logistic Regression Accuracy on Test Set:", lr_acc_test)
print("Logistic Regression Accuracy on Training Set:", lr_acc_train)
print("Logistic Regression Cross-Validation Scores:", cv_lr)
print("Logistic Regression Cross-Validation Mean Score:", cv_lr.mean())
print("Logistic Regression Cross-Validation Standard Deviation:", cv_lr.std())
```

```
Logistic Regression Accuracy on Test Set: 0.007518796992481203
Logistic Regression Accuracy on Training Set: 0.022653721682847898
Logistic Regression Cross-Validation Scores: [0.01123596 0.01123596 0.01136364 0.01136364 0.01136364]
Logistic Regression Cross-Validation Mean Score: 0.01131256384065373
Logistic Regression Cross-Validation Standard Deviation: 6.255081059201201e-05
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1 members, which is less than n_splits=5.
  warnings.warn(
```

Random Forest Classifier

In [5]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# Random Forest
rf = RandomForestClassifier()
rf.fit(x_train, y_train)

# Accuracy Scores
rf_acc_test = rf.score(x_test, y_test)
rf_acc_train = rf.score(x_train, y_train)

# Cross-Validation
cv_rf = cross_val_score(rf, x, y, cv=5)

# Printing Results
print("Random Forest Accuracy on Test Set:", rf_acc_test)
print("Random Forest Accuracy on Training Set:", rf_acc_train)
print("Random Forest Cross-Validation Scores:", cv_rf)
print("Random Forest Cross-Validation Mean Score:", cv_rf.mean())
print("Random Forest Cross-Validation Standard Deviation:", cv_rf.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
Random Forest Accuracy on Test Set: 0.0
Random Forest Accuracy on Training Set: 1.0
Random Forest Cross-Validation Scores: [0.01123596 0.01123596 0.02272727 0.         0.02272727]
Random Forest Cross-Validation Mean Score: 0.013585291113381002
Random Forest Cross-Validation Standard Deviation: 0.008517635236520864
```

GaussianNB

In [6]:
```python
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# Naive Bayes
nb = GaussianNB()
nb.fit(x_train, y_train)

# Accuracy Scores
nb_acc_test = nb.score(x_test, y_test)
nb_acc_train = nb.score(x_train, y_train)

# Cross-Validation
cv_nb = cross_val_score(nb, x, y, cv=5)

# Printing Results
print("Naive Bayes Accuracy on Test Set:", nb_acc_test)
print("Naive Bayes Accuracy on Training Set:", nb_acc_train)
print("Naive Bayes Cross-Validation Scores:", cv_nb)
print("Naive Bayes Cross-Validation Mean Score:", cv_nb.mean())
print("Naive Bayes Cross-Validation Standard Deviation:", cv_nb.std())
```

```
Naive Bayes Accuracy on Test Set: 0.0
Naive Bayes Accuracy on Training Set: 0.6504854368932039
Naive Bayes Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
Naive Bayes Cross-Validation Mean Score: 0.004545454545454545
Naive Bayes Cross-Validation Standard Deviation: 0.005567022142689042
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

KNeighbors Classifier

In [7]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# K-Nearest Neighbors (KNN)
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)

# Accuracy Scores
knn_acc_test = knn.score(x_test, y_test)
knn_acc_train = knn.score(x_train, y_train)

# Cross-Validation
cv_knn = cross_val_score(knn, x, y, cv=5)

# Printing Results
print("K-Nearest Neighbors Accuracy on Test Set:", knn_acc_test)
print("K-Nearest Neighbors Accuracy on Training Set:", knn_acc_train)
print("K-Nearest Neighbors Cross-Validation Scores:", cv_knn)
print("K-Nearest Neighbors Cross-Validation Mean Score:", cv_knn.mean())
print("K-Nearest Neighbors Cross-Validation Standard Deviation:", cv_knn.std())
```

```
K-Nearest Neighbors Accuracy on Test Set: 0.0
K-Nearest Neighbors Accuracy on Training Set: 0.1715210355987055
K-Nearest Neighbors Cross-Validation Scores: [0. 0. 0. 0. 0.]
K-Nearest Neighbors Cross-Validation Mean Score: 0.0
K-Nearest Neighbors Cross-Validation Standard Deviation: 0.0
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

AdaBoost Classifier

In [8]:
```python
from sklearn.ensemble import AdaBoostClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.7)

# AdaBoost
adaboost = AdaBoostClassifier()
adaboost.fit(x_train, y_train)

# Accuracy Scores
adaboost_acc_test = adaboost.score(x_test, y_test)
adaboost_acc_train = adaboost.score(x_train, y_train)

# Cross-Validation
cv_adaboost = cross_val_score(adaboost, x, y, cv=5)

# Printing Results
print("AdaBoost Accuracy on Test Set:", adaboost_acc_test)
print("AdaBoost Accuracy on Training Set:", adaboost_acc_train)
print("AdaBoost Cross-Validation Scores:", cv_adaboost)
print("AdaBoost Cross-Validation Mean Score:", cv_adaboost.mean())
print("AdaBoost Cross-Validation Standard Deviation:", cv_adaboost.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
AdaBoost Accuracy on Test Set: 0.0
AdaBoost Accuracy on Training Set: 0.03559870550161812
AdaBoost Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
AdaBoost Cross-Validation Mean Score: 0.004545454545454545
AdaBoost Cross-Validation Standard Deviation: 0.005567022142689042
```

With Ratio 80 - 20

SVM

In [15]:
```
from sklearn.svm import SVC
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# Support Vector Machine (SVM)
clf_svm = SVC()
clf_svm.fit(x_train, y_train)

# Accuracy Scores
svm_acc_test = clf_svm.score(x_test, y_test)
svm_acc_train = clf_svm.score(x_train, y_train)

# Cross-Validation
cv_svm = cross_val_score(clf_svm, x, y, cv=5)

# Printing Results
print("SVM Accuracy on Test Set:", svm_acc_test)
print("SVM Accuracy on Training Set:", svm_acc_train)
print("SVM Cross-Validation Scores:", cv_svm)
print("SVM Cross-Validation Scores percentage :", cv_svm * 100)
print("SVM Cross-Validation Mean Score:", cv_svm.mean())
print("SVM Cross-Validation Standard Deviation:", cv_svm.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1 members, which is less than n_splits=5.
  warnings.warn(
SVM Accuracy on Test Set: 0.011235955056179775
SVM Accuracy on Training Set: 0.14730878186968838
SVM Cross-Validation Scores: [0.01123596 0.01123596 0.02272727 0.02272727 0.02272727]
SVM Cross-Validation Scores percentage : [1.12359551 1.12359551 2.27272727 2.27272727 2.27272727]
SVM Cross-Validation Mean Score: 0.018130745658835545
SVM Cross-Validation Standard Deviation: 0.005629572953281053
```

Decision Tree Classifier

In [16]:
```
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)

# Accuracy Scores
dt_acc_test = dt.score(x_test, y_test)
dt_acc_train = dt.score(x_train, y_train)

# Cross-Validation
cv_dt = cross_val_score(dt, x, y, cv=5)

# Printing Results
print("Decision Tree Accuracy on Test Set:", dt_acc_test)
print("Decision Tree Accuracy on Training Set:", dt_acc_train)
print("Decision Tree Cross-Validation Scores:", cv_dt)
print("Decision Tree Cross-Validation Mean Score:", cv_dt.mean())
print("Decision Tree Cross-Validation Standard Deviation:", cv_dt.std())
```

```
Decision Tree Accuracy on Test Set: 0.0
Decision Tree Accuracy on Training Set: 1.0
Decision Tree Cross-Validation Scores: [0.         0.         0.01136364 0.         0.        ]
Decision Tree Cross-Validation Mean Score: 0.0022727272727272726
Decision Tree Cross-Validation Standard Deviation: 0.004545454545454545
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

Logistic Regression

In [9]:
```python
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# Logistic Regression
lr = LogisticRegression()
lr.fit(x_train, y_train)

# Accuracy Scores
lr_acc_test = lr.score(x_test, y_test)
lr_acc_train = lr.score(x_train, y_train)

# Cross-Validation
cv_lr = cross_val_score(lr, x, y, cv=5)

# Printing Results
print("Logistic Regression Accuracy on Test Set:", lr_acc_test)
print("Logistic Regression Accuracy on Training Set:", lr_acc_train)
print("Logistic Regression Cross-Validation Scores:", cv_lr)
print("Logistic Regression Cross-Validation Mean Score:", cv_lr.mean())
print("Logistic Regression Cross-Validation Standard Deviation:", cv_lr.std())
```

```
Logistic Regression Accuracy on Test Set: 0.0
Logistic Regression Accuracy on Training Set: 0.0169971671388102
Logistic Regression Cross-Validation Scores: [0.01123596 0.01123596 0.01136364 0.01136364 0.01136364]
Logistic Regression Cross-Validation Mean Score: 0.01131256384065373
Logistic Regression Cross-Validation Standard Deviation: 6.255081059201201e-05
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1 members, which is less than n_splits=5.
  warnings.warn(
```

Random Forest Classifier

In [10]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# Random Forest
rf = RandomForestClassifier()
rf.fit(x_train, y_train)

# Accuracy Scores
rf_acc_test = rf.score(x_test, y_test)
rf_acc_train = rf.score(x_train, y_train)

# Cross-Validation
cv_rf = cross_val_score(rf, x, y, cv=5)

# Printing Results
print("Random Forest Accuracy on Test Set:", rf_acc_test)
print("Random Forest Accuracy on Training Set:", rf_acc_train)
print("Random Forest Cross-Validation Scores:", cv_rf)
print("Random Forest Cross-Validation Mean Score:", cv_rf.mean())
print("Random Forest Cross-Validation Standard Deviation:", cv_rf.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1 members, which is less than n_splits=5.
  warnings.warn(
Random Forest Accuracy on Test Set: 0.0
Random Forest Accuracy on Training Set: 1.0
Random Forest Cross-Validation Scores: [0.01123596 0.         0.02272727 0.         0.01136364]
Random Forest Cross-Validation Mean Score: 0.009065372829417773
Random Forest Cross-Validation Standard Deviation: 0.008497092683589549
```

GaussianNB

In [11]:
```
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# Naive Bayes
nb = GaussianNB()
nb.fit(x_train, y_train)

# Accuracy Scores
nb_acc_test = nb.score(x_test, y_test)
nb_acc_train = nb.score(x_train, y_train)

# Cross-Validation
cv_nb = cross_val_score(nb, x, y, cv=5)

# Printing Results
print("Naive Bayes Accuracy on Test Set:", nb_acc_test)
print("Naive Bayes Accuracy on Training Set:", nb_acc_train)
print("Naive Bayes Cross-Validation Scores:", cv_nb)
print("Naive Bayes Cross-Validation Mean Score:", cv_nb.mean())
print("Naive Bayes Cross-Validation Standard Deviation:", cv_nb.std())
```

```
Naive Bayes Accuracy on Test Set: 0.0
Naive Bayes Accuracy on Training Set: 0.5949008498583569
Naive Bayes Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
Naive Bayes Cross-Validation Mean Score: 0.004545454545454545
Naive Bayes Cross-Validation Standard Deviation: 0.005567022142689042
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

KNeighbors Classifier

In [12]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# K-Nearest Neighbors (KNN)
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)

# Accuracy Scores
knn_acc_test = knn.score(x_test, y_test)
knn_acc_train = knn.score(x_train, y_train)

# Cross-Validation
cv_knn = cross_val_score(knn, x, y, cv=5)

# Printing Results
print("K-Nearest Neighbors Accuracy on Test Set:", knn_acc_test)
print("K-Nearest Neighbors Accuracy on Training Set:", knn_acc_train)
print("K-Nearest Neighbors Cross-Validation Scores:", cv_knn)
print("K-Nearest Neighbors Cross-Validation Mean Score:", cv_knn.mean())
print("K-Nearest Neighbors Cross-Validation Standard Deviation:", cv_knn.std())
```

```
K-Nearest Neighbors Accuracy on Test Set: 0.0
K-Nearest Neighbors Accuracy on Training Set: 0.17563739376770537
K-Nearest Neighbors Cross-Validation Scores: [0. 0. 0. 0. 0.]
K-Nearest Neighbors Cross-Validation Mean Score: 0.0
K-Nearest Neighbors Cross-Validation Standard Deviation: 0.0
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
```

AdaBoost Classifier

In [13]:
```
from sklearn.ensemble import AdaBoostClassifier
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split, cross_val_score

# Loading the dataset
x, y = load_diabetes(return_X_y=True)

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size=0.8)

# AdaBoost
adaboost = AdaBoostClassifier()
adaboost.fit(x_train, y_train)

# Accuracy Scores
adaboost_acc_test = adaboost.score(x_test, y_test)
adaboost_acc_train = adaboost.score(x_train, y_train)

# Cross-Validation
cv_adaboost = cross_val_score(adaboost, x, y, cv=5)

# Printing Results
print("AdaBoost Accuracy on Test Set:", adaboost_acc_test)
print("AdaBoost Accuracy on Training Set:", adaboost_acc_train)
print("AdaBoost Cross-Validation Scores:", cv_adaboost)
print("AdaBoost Cross-Validation Mean Score:", cv_adaboost.mean())
print("AdaBoost Cross-Validation Standard Deviation:", cv_adaboost.std())
```

```
C:\Users\Arooj\AppData\Roaming\Python\Python311\site-packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated class in y has only 1
members, which is less than n_splits=5.
  warnings.warn(
AdaBoost Accuracy on Test Set: 0.0
AdaBoost Accuracy on Training Set: 0.0339943342776204
AdaBoost Cross-Validation Scores: [0.         0.         0.01136364 0.01136364 0.        ]
AdaBoost Cross-Validation Mean Score: 0.004545454545454545
AdaBoost Cross-Validation Standard Deviation: 0.005567022142689042
```

In [ ]: