Classifier Class

January 1, 2020

1 Classifier Class

This class wraps 4 kinds of classifiers **naive_bayes, DecisionTreeClassifier, RandomForestClassifier, KNeighborsClassifier**

[1]:
```
import numpy as np
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

class Classifier:
    def __init__(self, method):
        if(method == 'BAYES'):
            self.clf = GaussianNB()
        elif(method == 'DTREE'):
            self.clf = DecisionTreeClassifier()
        elif(method == 'KNN'):
            self.clf = KNeighborsClassifier()
        elif(method == 'RF'):
            self.clf = RandomForestClassifier()
        else:
            #picking bayes as default classifier if input invalid
            self.clf = GaussianNB()

    def fit(self, X_train, y_train):
        self.clf.fit(X_train, y_train)

    def predict(self, X_test):
        return self.clf.predict(X_test)

    def score(self, X_test, y_test):
        return self.clf.score(X_test, y_test)
```

1.0.1 Parameters

method: String, (default = BAYES)

- BAYES: uses naive bayes classifier
- DTREE: uses decision tree classifier
- KNN: uses nearest neighbor classifier
- RF: uses random forest classifier

1.1 fit(self, X_train, y_train)

fit classifier according to X_train, y_train

1.1.1 Parameters

X_train: array-like, shape (n_samples, n_features)

- Training vectors, where n_samples is the number of samples and n_features is the number of features.

y_train: array-like, shape (n_samples)

- Target values.

1.2 predict(self, X_test)

perform classification on an array of test vectors X.

1.2.1 Parameters

X_test: array-like, shape (n_samples, n_features)

- vector to perform classification on.

returns: ndarray of shape (n_samples) predicted values for X_test.

1.3 score(self, X_test, y_test)

Returns the mean accuracy on the given test data and labels.

1.3.1 Parameters

X_test: array-like of shape (n_samples, n_features)

- Test samples.

y_test: array-like of shape (n_samples,) or (n_samples, n_outputs)

- True labels for X.

returns: float Mean accuracy of self.predict(X_test) wrt. y_test.

1.4 Example

[2]:
```
X, y = datasets.load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)
clf = Classifier('KNN')
clf.fit(X_train, y_train)
clf.predict(X_test)
```

[2]:
```
array([2, 1, 1, 2, 0, 2, 1, 1, 2, 1, 2, 0, 2, 2, 1, 0, 1, 0, 0, 2, 2, 1,
       1, 2, 2, 1, 1, 0, 0, 1, 0, 2, 0, 2, 1, 1, 1, 0, 1, 0, 0, 2, 0, 1,
       0, 1, 1, 2, 2, 2, 1, 1, 0, 0, 2, 2, 1, 1, 2, 0, 1, 1, 2, 1, 0, 2,
       0, 0, 2, 0, 1, 0, 1, 2, 0])
```

[3]:
```
clf.score(X_test, y_test)
```

[3]:
```
1.0
```