T.C.
ERCİYES ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

Diagnosis of Cancer Using Blood Microbiome Data

Hazırlayan
Duygu Gözde KAYABAŞI
1030510338

Bilgisayar Mühendisliği
Introduction to Machine Learning
Midterm Project

Mayıs
2024
KAYSERİ

```
import numpy
import pandas
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix

data = pandas.read_csv("data.csv")
#data.describe()
labels = pandas.read_csv("labels.csv")
#labels.describe()
data.drop_duplicates()
labels.drop_duplicates()
#data.columns.values[0] = 'Sample' I ALREADY DID THAT
#1836 different microorganisms appear as features.
input = data.drop('Sample', axis = 1)
#input
output = labels['disease_type']
#output
#train %75 test %25
X_train, X_test, y_train, y_test = train_test_split(input, output,
test_size = 0.25, random_state= 42)
y_train = y_train.values.ravel()
y_test = y_test.values.ravel()

random_forest = RandomForestClassifier(n_estimators = 100, max_depth =
3)
random_forest.fit(X_train ,y_train)
accuracy_random_forest = round(random_forest.score(X_train, y_train) *
100, 2)
accuracy_random_forest

prediction_from_random_forest = random_forest.predict(X_test)
accuracy_score_random_forest =
accuracy_score(prediction_from_random_forest, y_test)

#correct_first_class , total_first_class , correct_second_class ,
total_second_class
true_positive , false_positive , false_negative , true_negative =
confusion_matrix(y_test, prediction_from_random_forest)
sensitivity_random_forest = numpy.empty(0)
specificity_random_forest = numpy.empty(0)
```

```
sensitivity_random_forest = numpy.append(sensitivity_random_forest,
numpy.array([true_positive / (true_positive + false_negative)]))
specificity_random_forest = numpy.append(specificity_random_forest,
numpy.array([true_negative / (true_negative + false_positive)]))

print('Training Features Shape:', X_train.shape)
print('Training Labels Shape:', y_train.shape)
print('Testing Features Shape:', X_test.shape)
print('Testing Labels Shape:', y_test.shape)

print("RANDOM FOREST ALGORITHM SENSITIVITY IS:",
sensitivity_random_forest)
print("RANDOM FOREST ALGORITHM SPECIFICITY IS:",
specificity_random_forest)

print("RANDOM FOREST ALGORITHM ACCURACY SCORE")
print(round(accuracy_random_forest,2,), "%")

gradient_boosting = GradientBoostingClassifier(n_estimators = 100,
max_depth = 3)

gradient_boosting.fit(X_train, y_train)

accuracy_gradient = round(gradient_boosting.score(X_train, y_train) *
100, 2)

prediction_from_gradient_boost = gradient_boosting.predict(X_test)
accuracy_score_gradient_boost =
accuracy_score(prediction_from_gradient_boost, y_test)

true_negative, false_positive, false_negative, true_positive =
confusion_matrix(y_test, prediction_from_gradient_boost)
sensitivity_gradient_boost = numpy.empty(0)
specificity_gradient_boost = numpy.empty(0)

sensitivity_gradient_boost = numpy.append(sensitivity_gradient_boost,
numpy.array([true_positive / (true_positive + false_negative)]))
specificity_gradient_boost = numpy.append(specificity_gradient_boost,
numpy.array([true_negative / (true_negative + false_positive)]))

print("GRADIENT BOOSTED TREE ALGORITHM SENSITIVITY IS:",
sensitivity_gradient_boost)
```

```
print("GRADIENT BOOSTED TREE ALGORITHM SPECIFICITY IS:",
specificity_gradient_boost)

print("GRADIENT BOOSTED TREES ALGORITHM ACCURACY SCORE")
print(round(accuracy_gradient,2,), "%")

#Output
<ipython-input-19-c3e53ee0a2ad>:39: RuntimeWarning: invalid value
encountered in divide
  specificity_random_forest = numpy.append(specificity_random_forest,
numpy.array([true_negative / (true_negative + false_positive)]))
Training Features Shape: (266, 1836)
Training Labels Shape: (266,)
Testing Features Shape: (89, 1836)
Testing Labels Shape: (89,)
RANDOM FOREST ALGORITHM SENSITIVITY IS: [1. 1. 0. 1.]
RANDOM FOREST ALGORITHM SPECIFICITY IS: [       nan 0.         1.
0.96666667]
RANDOM FOREST ALGORITHM ACCURACY SCORE
93.61 %
GRADIENT BOOSTED TREE ALGORITHM SENSITIVITY IS: [ 1. nan  0.  1.]
GRADIENT BOOSTED TREE ALGORITHM SPECIFICITY IS: [ 1.  0. nan nan]
GRADIENT BOOSTED TREES ALGORITHM ACCURACY SCORE
100.0 %
<ipython-input-19-c3e53ee0a2ad>:65: RuntimeWarning: invalid value
encountered in divide
  sensitivity_gradient_boost = numpy.append(sensitivity_gradient_boost,
numpy.array([true_positive / (true_positive + false_negative)]))
<ipython-input-19-c3e53ee0a2ad>:66: RuntimeWarning: invalid value
encountered in divide
  specificity_gradient_boost = numpy.append(specificity_gradient_boost,
numpy.array([true_negative / (true_negative + false_positive)]))
```