Machine Learning

Assignment 2

Submitted By:

Yearagra Paliwal

(SP22004)

Question 3:

Pre-Processing: -

Dataset:

```
# DATASET

data = pd.read_csv('BitcoinHeistData.csv')
data = data.drop(['address'], axis=1)
data
```

Removes address column and printed the dataset.

Encoding:

```
# ENCODING

labelencoder_Y = LabelEncoder()
data['lable'] = labelencoder_Y.fit_transform(data['label'])
data
```

Encoded the label column and printed the dataset.

EDA Report:

```
# EDA REPORT

profile = ProfileReport(data, title="Data Analysis Report")
profile.to_notebook_iframe()
```

Generated EDA and analyse the data.

Train Validation Test Split:

```
# TRAIN TEST VALIDATION SPLIT

X= data.drop(columns='label')
Y=data[['label']]
def train_val_test_split(data, labels, train,val,test):
    print("Length: "+str(len(data)))

    train_data=data[:len(data)*0.7,:]
    train_labels=labels[:len(data)*0.7,:]

    val_data=data[len(data)*0.7:len(data)*0.85,:]
    val_labels=labels[len(data)*0.7:len(data)*0.85,:]

    test_data=data[len(data)*0.85:,:]
    test_labels=labels[len(data)*0.85:,:]

    return train_data,train_labels,val_data,val_labels,test_data,test_labels
```

```
# SPLITTED DATA
print(X_train.shape, X_val.shape, X_test.shape)
print(Y_train.shape, Y_val.shape, Y_test.shape)
```

```
(2041687, 8) (437504, 8) (437506, 8)
(2041687, 1) (437504, 1) (437506, 1)
```

Split the data into 3 parts train, validation, and test (70:15:15).

Part A: -

Gini Index:

```
# GINI INDEX

depth=[4,8,10,15,20]
trainingAcc=[]
testingAcc=[]

for i in depth:
    gini_tree = DecisionTreeClassifier(criterion='gini',max_depth=i)
    gini_tree = gini_tree.fit(X_train,Y_train)
    trainAcc = gini_tree.score(X_train,Y_train)
    testAcc = gini_tree.score(X_test,Y_test)
    print("Accuracy For Depth: ",i)
    print("Accuracy: ", trainAcc)
   # print("Accuracy test: ", testAcc)
    print()
    trainingAcc.append(trainAcc*100)
    testingAcc.append(testAcc*100)
```

Output:

```
Accuracy For Depth:  4
Accuracy:  0.9840328120813817

Accuracy For Depth:  8
Accuracy:  0.9854747569044618

Accuracy For Depth:  10
Accuracy:  0.9861198117047324

Accuracy For Depth:  15
Accuracy:  0.9886549701300934

Accuracy For Depth:  20
Accuracy:  0.9922377915909736
```

Entropy:

```
# ENTROPY

depth=[4,8,10,15,20]
trainingAcc=[]
testingAcc=[]

for i in depth:
    entropy_tree = DecisionTreeClassifier(criterion='entropy',max_depth=i)
    entropy_tree.fit(X_train,Y_train)
    trainAcc = entropy_tree.score(X_train,Y_train)
    testAcc = entropy_tree.score(X_test,Y_test)
    print("Accuracy For Depth: ",i)
    print("Accuracy: ", trainAcc)
    print()
    trainingAcc.append(trainAcc*100)
    testingAcc.append(testAcc*100)
```

Output:

```
Accuracy For Depth:  4
Accuracy:  0.9840847299316693

Accuracy For Depth:  8
Accuracy:  0.9854556550538843

Accuracy For Depth:  10
Accuracy:  0.9860247922428854

Accuracy For Depth:  15
Accuracy:  0.9889380693514725

Accuracy For Depth:  20
Accuracy:  0.9926261958860492
```

(So, Entropy gives better accuracy)

Part B: -

50% DATASET RANDOMLY AND MAKING 100 DECISION TREES (MAX DEAPTH 3)

```
# GETTING 50% DATASET RANDOMLY AND MAKING 100 DECISION TREES WITH (MAX DEAPTH 3)

def max_vote(predictions):
    final_prediction=[]
    for j in range(len(predictions[0])):
        maxi={}
        for i in predictions:
            if(i[j] in maxi):
                maxi[i[j]]+=1
            else:
                maxi[i[j]]=1
        max_key=max(maxi, key= lambda X: maxi[X])
        final_prediction.append(max_key)

    return final_prediction

stumps=[]
predictions=[]
for i in range(100):
    stumps.append(DecisionTreeClassifier(criterion="entropy",max_depth=3))
    X_train_frac=X_train.sample(frac=0.5)
    Y_train_frac=Y_train.loc[X_train_frac.index]
    stumps[i].fit(X_train,Y_train)
    predicts=stumps[i].predict(X_test)
    predictions.append(predicts)

final_prediction=max_vote(predictions)
accuracy=np.sum(np.array(final_prediction)==np.array(Y_test.to_list()))/len(Y_test)

print("Accuracy: "+str(accuracy))
```

Output:

```
Accuracy: 98.52815889526086 %
```

(Because the stumps are weak and only have a depth of 3, the accuracy is lower than in part a)

Part C: -

Adaboost:

```
# ADABOOST

estimators = [4, 8, 10, 15, 20]
testAcc=[]
trainAcc=[]
valAcc=[]

for i in estimators:
    adaboost_tree = AdaBoostClassifier(n_estimators=i, base_estimator=DecisionTreeClassifier(criterion="entropy", max_depth=i))
    adaboost_tree.fit(X_train,Y_train)
    testAcc.append(adaboost_tree.score(X_test,Y_test))
    trainAcc.append(adaboost_tree.score(X_train,Y_train))
    valAcc.append(adaboost_tree.score(X_val,Y_val))
    print("Accuracy For Estimate: ",i)
    print("Training Accuracy: ", trainAcc[-1])
    print("Validating Accuracy: ", valAcc[-1])
    print("Testing Accuracy: ", testAcc[-1])
```

Output:

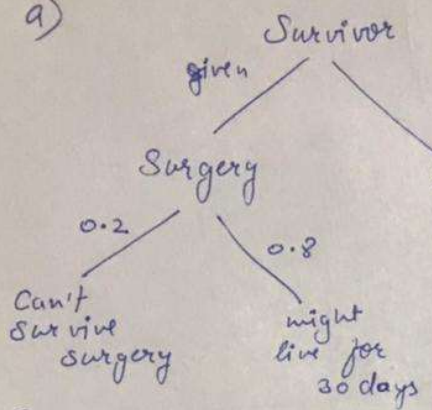
```
Accuracy For Estimate:  4
Training Accuracy:  0.9774064290951552
Validating Accuracy:  2.0571240491515506e-05
Testing Accuracy:  5.9427756419340534e-05

Accuracy For Estimate:  8
Training Accuracy:  0.9386213459751667
Validating Accuracy:  0.0007588502047981276
Testing Accuracy:  0.005803348982642524

Accuracy For Estimate:  10
Training Accuracy:  0.9481859854130432
Validating Accuracy:  0.009741625219426566
Testing Accuracy:  0.02413681183800908

Accuracy For Estimate:  15
Training Accuracy:  0.9896967556731272
Validating Accuracy:  0.025138055880631 95
Testing Accuracy:  0.027821332736008193

Accuracy For Estimate:  20
Training Accuracy:  0.9995361678846953
Validating Accuracy:  0.0992653781451141
Testing Accuracy:  0.09312329430910662
```

(After that, when max depth is used, overfitting begins, training accuracy reaches 1, and testing and validation do not improve.)

(So, the accuracy is more better in Random Forest as compared to adaboost.)

Question 1:

a)



So, the probality that the person can't survive surgery is is 0.2.

And, the Probolity that the person might live for 30 days after a surgery is 0.8.

c) Surgery → survive → +ve (Positive)

① $\left(\frac{\text{+ve test}}{\text{survive surgery}}\right) = 0.95$

② $\left(\frac{\text{+ve test}}{\text{can't survive}}\right) = 0.05$

③ $\left(\frac{\text{Survive surgery}}{\text{+ve test}}\right) = ?$

So, $= \left(\frac{\text{+ve test}}{\text{survive surgery}}\right)\left(\frac{(\text{survive surgery})}{(\text{+ve test})}\right)$

$= 0.95 \times \frac{0.8}{1}$

$= \boxed{0.76}$

d) If the result of test given +ve, we can identify the chances of survival which came out to be 0.98

b) $\angle(3) = 0.8$

$\angle(30) = 1.0$

$\angle(0) = 0$

Patient lived for 3 days with surgery performed

$\angle(n) = ?$

So, = $\frac{0.8}{1.0} \times$

Thank You