Practical No. 8

Title : Implementation of Boosting Algorithms

Implementation

Program and Output :

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
import pandas as pd
# Import train_test_split function
from sklearn.model_selection import train_test_split
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
```

```
# Load in the data
url='https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/mushrooms.csv'
df=pd.read_csv(url)
df
```

	type	cap_shape	cap_surface	cap_color	bruises	odor	gill_attachment	gill_spacing	gill_size	gill_color	...	stalk_surface_below_ring	stalk_color_above_ring	stalk_color_below_ring	veil_type	veil_color	ring_number	ring_type	spore_print_color	population	habitat
0	p	x	s	n	t	p	f	c	n	k	...	s	w	w	p	w	o	p	k	s	u
1	e	x	s	y	t	a	f	c	b	k	...	s	w	w	p	w	o	p	n	n	g
2	e	b	s	w	t	l	f	c	b	n	...	s	w	w	p	w	o	p	n	n	m
3	p	x	y	w	t	p	f	c	n	n	...	s	w	w	p	w	o	p	k	s	u
4	e	x	s	g	f	n	f	w	b	k	...	s	w	w	p	w	o	e	n	a	g
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8119	e	k	s	n	f	n	a	c	b	y	...	s	o	o	p	o	o	p	b	c	l
8120	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	n	o	p	b	v	l
8121	e	f	s	n	f	n	a	c	b	n	...	s	o	o	p	o	o	p	b	c	l
8122	p	k	y	n	f	y	f	c	n	b	...	k	w	w	p	w	o	e	w	v	l
8123	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	o	o	p	o	c	l

8124 rows × 23 columns

```
#Define the column names
df.columns = ['target','cap-shape','cap-surface','cap-color','bruises','odor','gill-attachment','gill-spacing',
'gill-size','gill-color','stalk-shape','stalk-root','stalk-surface-above-ring','stalk-surface-below-ring','stalk-color-above-ring',
'stalk-color-below-ring','veil-type','veil-color','ring-number','ring-type','spore-print-color','population',
'habitat']
for label in df.columns:
 df[label] = LabelEncoder().fit(df[label]).transform(df[label])
```

```
#Display information about the data set
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8124 entries, 0 to 8123
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   target                    8124 non-null   int64
 1   cap-shape                 8124 non-null   int64
 2   cap-surface               8124 non-null   int64
 3   cap-color                 8124 non-null   int64
 4   bruises                   8124 non-null   int64
 5   odor                      8124 non-null   int64
 6   gill-attachment           8124 non-null   int64
 7   gill-spacing              8124 non-null   int64
 8   gill-size                 8124 non-null   int64
 9   gill-color                8124 non-null   int64
 10  stalk-shape               8124 non-null   int64
 11  stalk-root                8124 non-null   int64
 12  stalk-surface-above-ring  8124 non-null   int64
 13  stalk-surface-below-ring  8124 non-null   int64
 14  stalk-color-above-ring    8124 non-null   int64
 15  stalk-color-below-ring    8124 non-null   int64
 16  veil-type                 8124 non-null   int64
 17  veil-color                8124 non-null   int64
 18  ring-number               8124 non-null   int64
 19  ring-type                 8124 non-null   int64
 20  spore-print-color         8124 non-null   int64
 21  population                8124 non-null   int64
 22  habitat                   8124 non-null   int64
dtypes: int64(23)
memory usage: 1.4 MB
None
```

```
X = df.drop(['target'], axis=1)
Y = df['target']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.3)

model = DecisionTreeClassifier(criterion='entropy', max_depth=1)
AdaBoost = AdaBoostClassifier(base_estimator=model, n_estimators=400,
learning_rate=1)

#Fit the model with training data
boostmodel = AdaBoost.fit(X_train, Y_train)
#Evaluate the accuracy of the model
y_pred = boostmodel.predict(X_test)
predictions = metrics.accuracy_score(Y_test, y_pred)
#Calculating the accuracy in percentage
print('The accuracy is: ', predictions * 100, '%')
```

```
The accuracy is:  100.0 %
```