

Practical 7

Implementation of Bagging Algo:

Program with Output:

```
#import Libraries
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, classification_report, ac
curacy_score
```

```
✓ 0s #import Libraries
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
```

```
#import dataset
data=pd.read_csv('Social_Network.csv')
data.head()
```

```
✓ 0s #import dataset
data=pd.read_csv('Social_Network.csv')
data.head()
```

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

```
#Extracting Independent and dependent variable
x = data.iloc[:, [2,3]].values
y = data.iloc[:,4].values
```

Kawar Nilesh Ramesh
Roll no : 59
Batch A

```
✓ [3] #Extracting Independent and dependent variable  
0s x = data.iloc[:,[2,3]].values  
y = data.iloc[:,4].values
```

```
#splitting the dataset into trainging ans testing set  
from sklearn.model_selection import train_test_split  
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0  
.25, random_state = 0)
```

```
✓ [4] #splitting the dataset into trainging ans testing set  
0s from sklearn.model_selection import train_test_split  
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)
```

```
#feature scaling  
from sklearn.preprocessing import StandardScaler  
st_x = StandardScaler()  
x_train = st_x.fit_transform(x_train)  
x_test = st_x.transform(x_test)
```

```
✓ [5] #feature scaling  
0s from sklearn.preprocessing import StandardScaler  
st_x = StandardScaler()  
x_train = st_x.fit_transform(x_train)  
x_test = st_x.transform(x_test)
```

```
#fitting the decision tree classifire  
from sklearn.tree import DecisionTreeClassifier  
classifier = DecisionTreeClassifier(criterion='entropy', random_state =  
0)  
classifier.fit(x_train, y_train)
```

```
✓ [6] #fitting the decision tree classifire  
0s from sklearn.tree import DecisionTreeClassifier  
classifier = DecisionTreeClassifier(criterion='entropy', random_state = 0)  
classifier.fit(x_train, y_train)  
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

```
#Predicting the test result
```

Kawar Nilesh Ramesh
Roll no : 59
Batch A

```
y_pred = classifier.predict(x_test)
y_predtrain = classifier.predict(x_train)
print(y_pred)
print("\n",y_predtrain)
```

```
✓ 0s #Predicting the test result
y_pred = classifier.predict(x_test)
y_predtrain = classifier.predict(x_train)
print(y_pred)
print("\n",y_predtrain)

[0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 1 1 1 0 0 1 0 0 1 0 1 0 1 0 0 0 1 1 0 0 1
 0 0 0 0 1 1 1 1 0 0 1 0 0 1 1 0 0 1 0 0 0 1 0 1 1 1]

[0 1 0 1 1 1 0 0 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 1 0 0 1 1 1 1 0 1 0 1 0 0 1
 0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 0 1
 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 1 1 1 0 0 0 0 0 1 0 0 1 1 1 1 1 0 1 1 0
 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0 1 0 0 0 1 1 0 0
 0 0 1 0 1 0 0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 0 0
 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0 0 0 0 0
 0 1 1 0 0 0 0 1 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 1 1 0 0 0
 0 0 1 0 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1
 0 0 0 0]
```

```
#confusion matrix for test set
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix for test data\n', cm)

#confusion matrix for train set
cm1=confusion_matrix(y_train,y_predtrain)
print('\nconfusion matrix for train data\n', cm1)

#Accuracy score for train and test set
print('\nAccuracy score for test data\n', accuracy_score(y_test,y_pred)
)
print('\nAccuracy Score for train data\n', accuracy_score(y_train,y_pre
dtrain))
```

Kawar Nilesh Ramesh
Roll no : 59
Batch A

```
✓ 0s #confusion matrix for test set
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix for test data\n', cm)

#confusion matrix for train set
cm1=confusion_matrix(y_train,y_predtrain)
print('\nconfusion matrix for train data\n', cm1)

#Accuracy score for train and test set
print('\nAccuracy score for test data\n', accuracy_score(y_test,y_pred))
print('\nAccuracy Score for train data\n', accuracy_score(y_train,y_predtrain))
```

confusion matrix for test data
[[62 6]
[3 29]]

confusion matrix for train data
[[189 0]
[0 111]]

Accuracy score for test data
0.91

Accuracy Score for train data
1.0

```
#classification report for train and test set
print('classification report for test data \n', classification_report(y_test,y_pred))
print('classification report for train data\n', classification_report(y_train,y_predtrain))
```

```
✓ 0s [9] #classification report for train and test set
print('classification report for test data \n', classification_report(y_test,y_pred))
print('classification report for train data\n', classification_report(y_train,y_predtrain))
```

classification report for test data

	precision	recall	f1-score	support
0	0.95	0.91	0.93	68
1	0.83	0.91	0.87	32
accuracy			0.91	100
macro avg	0.89	0.91	0.90	100
weighted avg	0.91	0.91	0.91	100

classification report for train data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	189
1	1.00	1.00	1.00	111
accuracy			1.00	300
macro avg	1.00	1.00	1.00	300
weighted avg	1.00	1.00	1.00	300

Kawar Nilesh Ramesh
Roll no : 59
Batch A