Practical 7

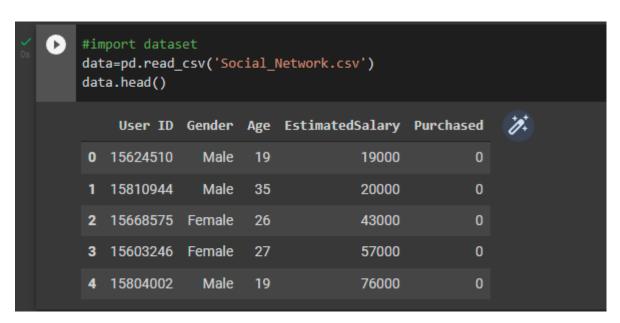
Implementation of Bagging Algo:

Program with Output:

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

#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()
```



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

```
[3] #Extracting Independent and dependent variable
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 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
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)
```

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

DecisionTreeClassifier(criterion='entropy', random_state=0)
```

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

```
#Predicting the test result
 y_pred = classifier.predict(x_test)
 y_predtrain = classifier.predict(x_train)
 print(y pred)
 print("\n",y_predtrain)
[000000001000001011010010100000000010000
 00001111001001100100010111]
 [0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1
 0010100010000111000000111110100000100
 0 0 0 01
```

```
#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))
```

```
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)
    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))
```

```
[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
                             0.91
                      0.95
                                         0.93
                                                    68
                      0.83
                               0.91
                                        0.87
                                         0.91
                                                   100
        accuracy
       macro avg
                      0.89
                             0.91
                                         0.90
                                                   100
                      0.91
                              0.91
                                         0.91
                                                   100
    weighted avg
    classification report for train data
                  precision recall f1-score support
                      1.00
                               1.00
                                         1.00
                                                   189
              0
                      1.00
                               1.00
                                         1.00
                                                    300
        accuracy
                                         1.00
       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