```
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier as dtc
from sklearn.ensemble import RandomForestClassifier as rfc
from sklearn.metrics import r2_score, classification_report as cr,
confusion_matrix as cm
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import numpy as np
```

```
z = pd.read_csv(r"C:\Users\skj_h\OneDrive\Desktop\
Social_Network_Ads.csv")
z
```

```
     Age  EstimatedSalary  Purchased
0     19            19000          0
1     35            20000          0
2     26            43000          0
3     27            57000          0
4     19            76000          0
..   ...              ...        ...
395   46            41000          1
396   51            23000          1
397   50            20000          1
398   36            33000          0
399   49            36000          1

[400 rows x 3 columns]
```

```
z.isnull().sum()
```

```
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

```
z.shape
```

```
(400, 3)
```

```
z.size
```

```
1200
```

```
z.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Age              400 non-null    int64
 1   EstimatedSalary  400 non-null    int64
 2   Purchased        400 non-null    int64
dtypes: int64(3)
memory usage: 9.5 KB
```

```
z.dtypes
```

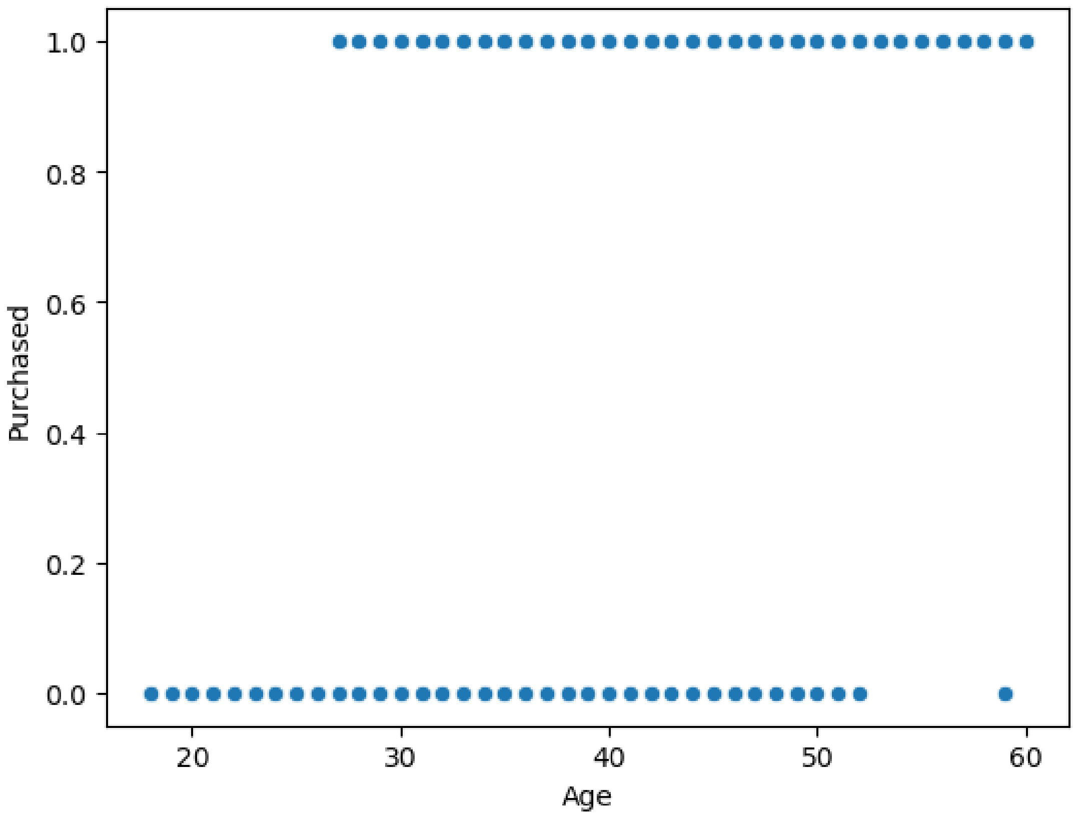
```
Age                int64
EstimatedSalary    int64
Purchased          int64
dtype: object
```

```
z.corr()
```

```
                      Age  EstimatedSalary  Purchased
Age              1.000000         0.155238   0.622454
EstimatedSalary  0.155238         1.000000   0.362083
Purchased        0.622454         0.362083   1.000000
```

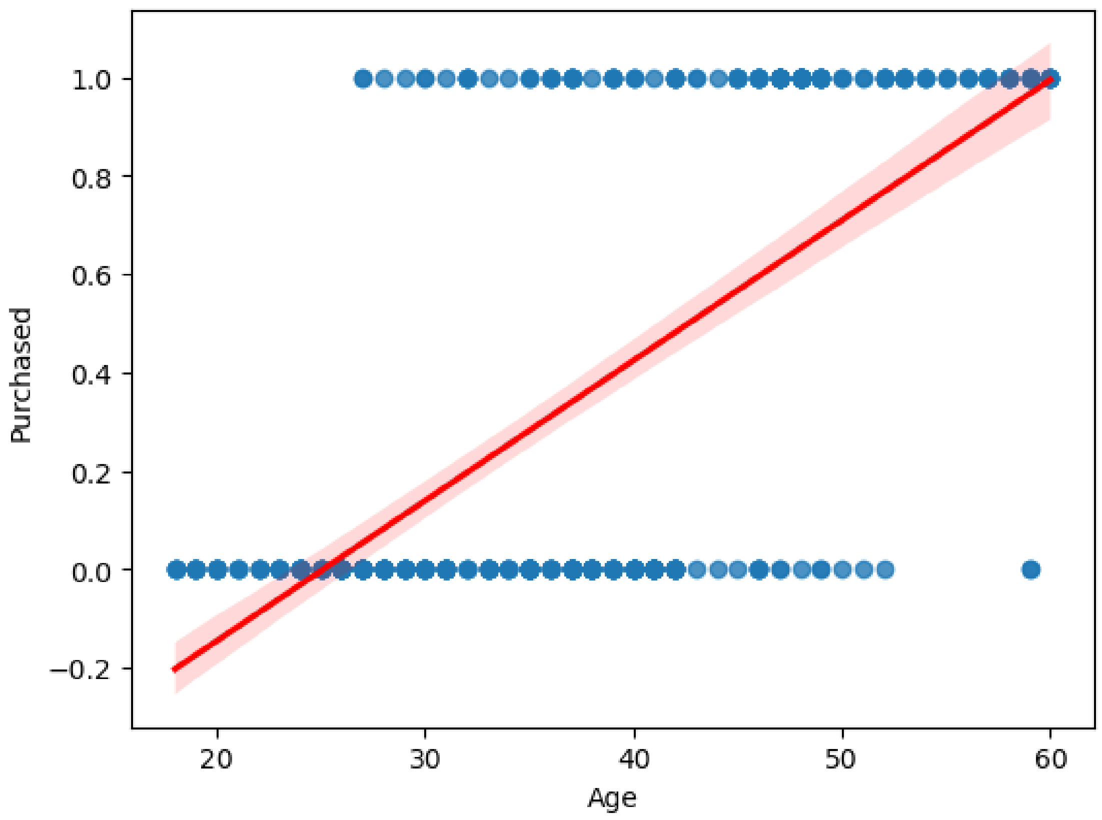
```
sns.scatterplot(x = z["Age"], y = z["Purchased"], data = z)
```

```
<Axes: xlabel='Age', ylabel='Purchased'>
```



```
sns.regplot(x = z["Age"], y = z["Purchased"], data = z, line_kws =
{"color": "red"})
```

```
<Axes: xlabel='Age', ylabel='Purchased'>
```



```
x = z[["Age", "Purchased"]]

X = x
Y = x["Purchased"]

x_train, x_test, y_train, y_test = train_test_split(X, Y, train_size =
0.7, test_size = 0.3, random_state = 100)

x_train = x_train.drop(["Purchased"], axis = 1)
x_test = x_test.drop(["Purchased"], axis = 1)

y_train = np.array(y_train).reshape(-1, 1)
y_test = np.array(y_test).reshape(-1, 1)

n = LogisticRegression()
n.fit(x_train, y_train)
```

```
LogisticRegression()
```

```
y_predict_train = n.predict(x_train)
r2_score(y_true = y_train, y_pred = y_predict_train)
```

```
0.26216640502354793
```

```
n = LogisticRegression()
n.fit(x_test, y_test)
```

```
LogisticRegression()
```

```
y_predict_test = n.predict(x_test)
r2_score(y_true = y_test, y_pred = y_predict_test)
```

```
0.39555555555555555
```

```
cr_train_logistic = cr(y_true = y_train, y_pred = y_predict_train)
```

```
cr_test_logistic = cr(y_true = y_test, y_pred = y_predict_test)
```

```
a = dtc()
a.fit(x_train, y_train)
```

```
DecisionTreeClassifier()
```

```
y_predict_train1 = a.predict(x_train)
cr_train_decision = cr(y_true = y_train, y_pred = y_predict_train)
```

```
a = dtc()
a.fit(x_test, y_test)
```

```
DecisionTreeClassifier()
```

```
y_predict_test1 = a.predict(x_test)
cr_test_decision  = cr(y_true = y_test, y_pred = y_predict_test1)
```

```
y5 = rfc(n_estimators = 100)
y5.fit(x_train, y_train)
```

```
RandomForestClassifier()
```

```
y_predict_train2 = y5.predict(x_train)
cr_train_random = cr(y_true = y_train, y_pred = y_predict_train2)
```

```
y5 = rfc()
y5.fit(x_test, y_test)
```

```
RandomForestClassifier()
```

```
y_predict_test2 = y5.predict(x_test)
cr_test_random = cr(y_true = y_test, y_pred = y_predict_test2)
```

```
def upper_name(s):
    return(s.upper())
```

```
print(upper_name("Logistic Regression training"),cr_train_logistic)
print(upper_name("Decision  Tree classifier
training"),cr_train_decision)
print(upper_name("Random forest classifier training"),cr_train_random)
print(upper_name("Logistic Regression testing"),cr_test_logistic)
print(upper_name("Decision Tree classifier testing"),cr_test_decision)

print(upper_name("Random forest classifier testing"),cr_test_random)
```

```
LOGISTIC REGRESSION TRAINING               precision    recall  f1-
score   support

           0       0.84      0.92      0.88       182
           1       0.81      0.67      0.74        98

    accuracy                           0.83       280
   macro avg       0.83      0.80      0.81       280
weighted avg       0.83      0.83      0.83       280

DECISION  TREE CLASSIFIER TRAINING               precision    recall
f1-score   support

           0       0.84      0.92      0.88       182
           1       0.81      0.67      0.74        98

    accuracy                           0.83       280
   macro avg       0.83      0.80      0.81       280
weighted avg       0.83      0.83      0.83       280

RANDOM FOREST CLASSIFIER TRAINING               precision    recall
f1-score   support

           0       0.84      0.85      0.84       182
           1       0.72      0.69      0.70        98

    accuracy                           0.80       280
   macro avg       0.78      0.77      0.77       280
weighted avg       0.80      0.80      0.80       280

LOGISTIC REGRESSION TESTING               precision    recall  f1-
score   support

           0       0.84      0.96      0.89        75
           1       0.91      0.69      0.78        45

    accuracy                           0.86       120
   macro avg       0.87      0.82      0.84       120
weighted avg       0.87      0.86      0.85       120

DECISION TREE CLASSIFIER TESTING               precision    recall
f1-score   support
```

```
              0       0.84      0.97      0.90        75
              1       0.94      0.69      0.79        45

       accuracy                           0.87       120
      macro avg       0.89      0.83      0.85       120
   weighted avg       0.88      0.87      0.86       120

RANDOM FOREST CLASSIFIER TESTING                precision    recall
f1-score   support

              0       0.86      0.93      0.90        75
              1       0.87      0.76      0.81        45

       accuracy                           0.87       120
      macro avg       0.87      0.84      0.85       120
   weighted avg       0.87      0.87      0.86       120
```