Import necessary libraries

In [1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusio
import matplotlib.pyplot as plt
import seaborn as sns
```

Load the dataset

In [3]:
```
df = pd.read_csv('https://github.com/sakshi2k/Social_Network_Ads/raw/master
```

Display the first few rows of the dataset to understand its structure

In [4]:
```
print(df.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
```

Select features and target variable

In [5]:
```
X = df[['Age', 'EstimatedSalary']]
y = df['Purchased']
```

Split the data into training and testing sets

In [6]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ra
```

Standardize the features (optional, but often improves performance)

In [7]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Create a Logistic Regression model

In [8]:
```
model = LogisticRegression(random_state=42)
```

Train the model

In [9]:
```
model.fit(X_train, y_train)
```

```
F:\FDriveSoftwares\programs\anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

Out[9]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=42, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

Make predictions on the test set

In [10]:
```
y_pred = model.predict(X_test)
```

Evaluate the model

In [11]:
```
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
```

```
Accuracy: 0.875
```

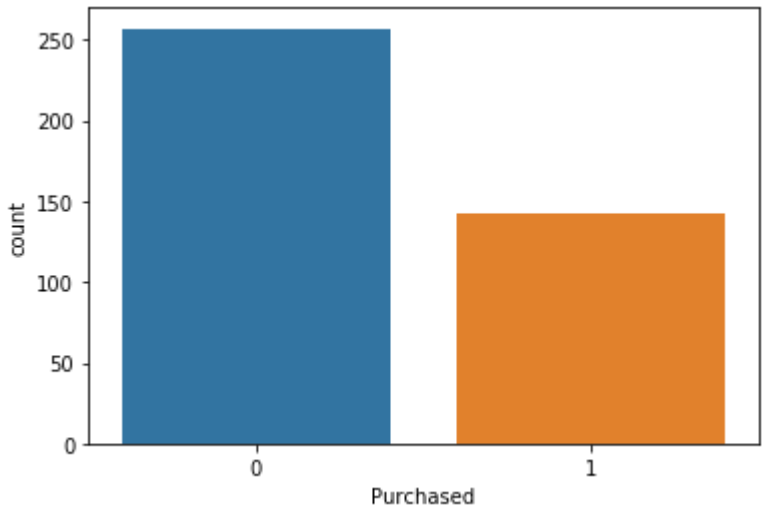
Display classification report and confusion matrix

In [19]:
```
print('Classification Report:\n', classification_report(y_test, y_pred))

# Create confusion matrix
conf_mat = confusion_matrix(y_test, y_pred)

print('Confusion Matrix:\n', conf_mat)
```

```
Classification Report:
               precision    recall  f1-score   support

           0       0.86      0.96      0.91        52
           1       0.91      0.71      0.80        28

    accuracy                           0.88        80
   macro avg       0.89      0.84      0.85        80
weighted avg       0.88      0.88      0.87        80

Confusion Matrix:
 [[50  2]
 [ 8 20]]
```

In [31]:
```
sns.countplot(x=y)
```

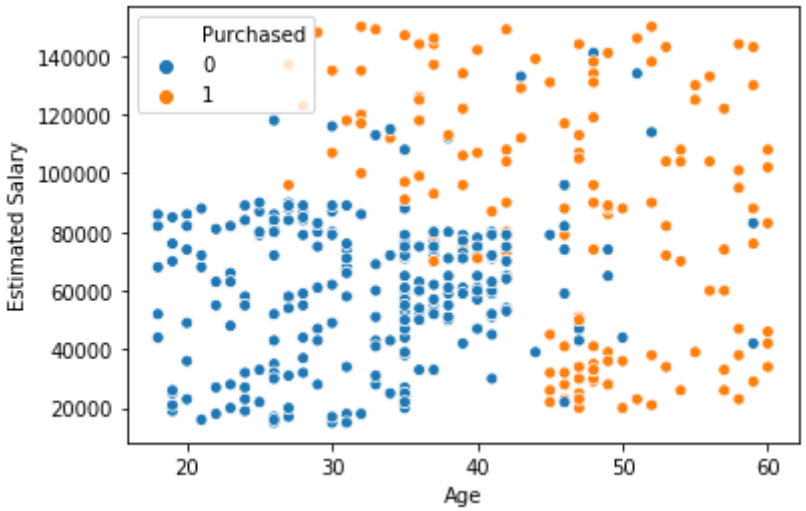
Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2a52f9a95c8>
```



Visualize the decision boundary (for two features)

In [13]:
```
sns.scatterplot(x='Age', y='EstimatedSalary', hue='Purchased', data=df)
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
```

Out[13]:
```
Text(0, 0.5, 'Estimated Salary')
```



In [ ]:

In [ ]:

In [ ]: