In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv("C:/Users/ameya/OneDrive/Desktop/DSBDAL/Social_Network_Ads.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	User ID	Gender	Age	EstimatedSalary	Purchased	Unnamed: 5
0	15624510	Male	19	19000	0	NaN
1	15810944	Male	35	20000	0	NaN
2	15668575	Female	26	43000	0	NaN
3	15603246	Female	27	57000	0	NaN
4	15804002	Male	19	76000	0	NaN

In [4]:

```
df2 = df.iloc[:,:-1]
```

In [6]:

```
df2.head()
```

Out[6]:

	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

In [7]:

```
df2.describe()
```

Out[7]:

	User ID	Age	EstimatedSalary	Purchased
count	4.000000e+02	400.000000	400.000000	400.000000
mean	1.569154e+07	37.655000	69742.500000	0.357500
std	7.165832e+04	10.482877	34096.960282	0.479864
min	1.556669e+07	18.000000	15000.000000	0.000000
25%	1.562676e+07	29.750000	43000.000000	0.000000
50%	1.569434e+07	37.000000	70000.000000	0.000000
75%	1.575036e+07	46.000000	88000.000000	1.000000
max	1.581524e+07	60.000000	150000.000000	1.000000

In [8]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   User ID          400 non-null    int64
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64
 3   EstimatedSalary  400 non-null    int64
 4   Purchased        400 non-null    int64
dtypes: int64(4), object(1)
memory usage: 15.8+ KB
```

In [9]:

```
df2['Purchased'].value_counts()
```

Out[9]:

```
0    257
1    143
Name: Purchased, dtype: int64
```

In [10]:

```
df2.isnull().sum()
```

Out[10]:

```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

In [11]:

```
df2['EstimatedSalary'].value_counts()
```

Out[11]:

```
72000     12
80000     11
79000     10
75000      9
71000      9
          ..
123000     1
37000      1
115000     1
148000     1
139000     1
Name: EstimatedSalary, Length: 117, dtype: int64
```

In [12]:

```
df2['Gender'].value_counts()
```

Out[12]:

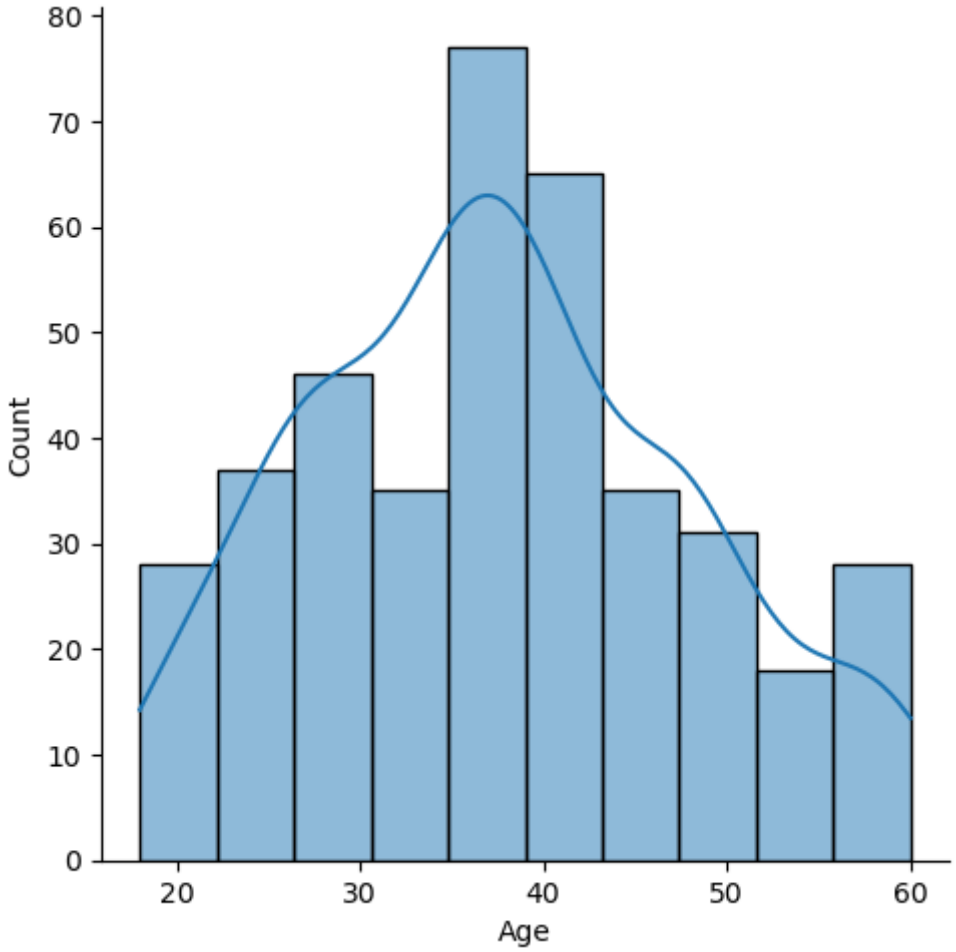
```
Female    204
Male      196
Name: Gender, dtype: int64
```

In [16]:

```
sns.displot(data = df2,x='Age',kde=True)
```

Out[16]:

```
<seaborn.axisgrid.FacetGrid at 0x18f29bb4550>
```

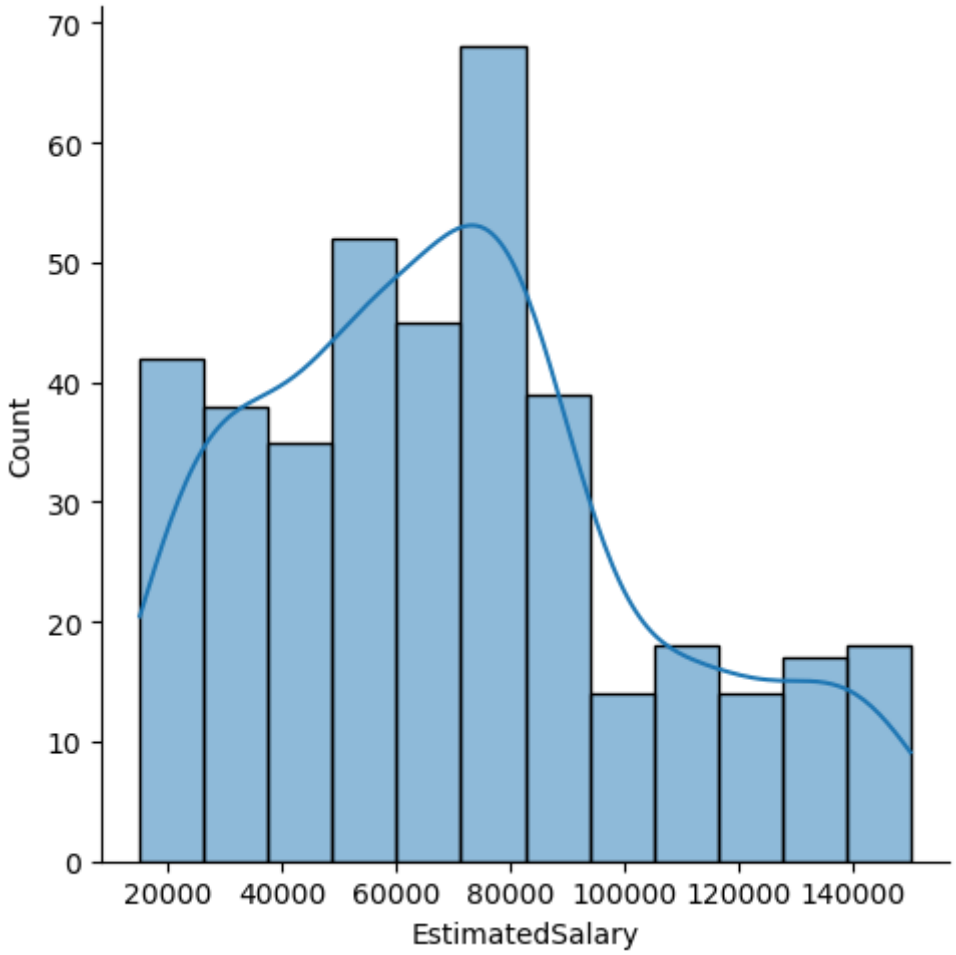


In [17]:

```
sns.displot(df['EstimatedSalary'], kde=True)
```

Out[17]:

```
<seaborn.axisgrid.FacetGrid at 0x18f284d1b20>
```



In [18]:

```
x = df2.iloc[:,2:4].values
y = df2.iloc[:,-1].values
```

In [19]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

Normalizing the dataset

In [20]:

```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
x_train = scalar.fit_transform(x_train)
x_test = scalar.fit_transform(x_test)
```

Model

In [21]:

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[21]:

```
LogisticRegression()
```

In [22]:

```
y_pred = classifier.predict(x_test)
```

In [23]:

```
y_pred
```

Out[23]:

```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1,
       0, 1, 0, 1, 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, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1], dtype=int64)
```

In [24]:

```
print("Training Accuracy: ",classifier.score(x_train,y_train)*100)
print("Testing Accuracy: ",classifier.score(x_test,y_test)*100)
```

```
Training Accuracy:  82.1875
Testing Accuracy:  88.75
```

Computing Accuracy

In [25]:

```
from sklearn.metrics import accuracy_score
Acc = accuracy_score(y_true=  y_test, y_pred = y_pred)
```

In [26]:

```
Acc
```

Out[26]:

```
0.8875
```

In [27]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_true=y_test,y_pred = y_pred)
print(cm)
```

```
[[52  6]
 [ 3 19]]
```

In [28]:

```
from sklearn.metrics import precision_score
ps = precision_score(y_test,y_pred)
```

In [29]:

```
ps
```

Out[29]:

```
0.76
```

In [30]:

```
from sklearn.metrics import precision_recall_fscore_support
prfc = precision_recall_fscore_support(y_test,y_pred)
print('precision:',prfc[0])
print('Recall:',prfc[1])
print('fscore:',prfc[2])
print('support:',prfc[3])
```

```
precision: [0.94545455 0.76      ]
Recall: [0.89655172 0.86363636]
fscore: [0.92035398 0.80851064]
support: [58 22]
```

In [2]:

```
#error rate = (fn+fp)/(tp + tn + fn + fp)
error_rate = 9/80
```

In [31]:

```
from sklearn.metrics import classification_report
cr = classification_report(y_test,y_pred)
print(cr)
```

```
              precision    recall  f1-score   support

           0       0.95      0.90      0.92        58
           1       0.76      0.86      0.81        22

    accuracy                           0.89        80
   macro avg       0.85      0.88      0.86        80
weighted avg       0.89      0.89      0.89        80
```

In [3]:

```
error_rate
```

Out[3]:

```
0.1125
```

In [ ]:

```

```