In [88]:

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

In [89]:

```
df= pd.read_csv('Social_Network_Ads.csv')
```

In [90]:

```
df.shape
```

Out[90]:

```
(400, 5)
```

In [91]:

```
df.head()
```

Out[91]:

	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 [92]:

```
df.drop(['User ID'],axis=1,inplace=True)
```

In [93]:

```
df.head()
```

Out[93]:

	Gender	Age	EstimatedSalary	Purchased
0	Male	19	19000	0
1	Male	35	20000	0
2	Female	26	43000	0
3	Female	27	57000	0
4	Male	19	76000	0

In [94]:

```
df.describe()
```

Out[94]:

	Age	EstimatedSalary	Purchased
count	400.000000	400.000000	400.000000
mean	37.655000	69742.500000	0.357500
std	10.482877	34096.960282	0.479864
min	18.000000	15000.000000	0.000000
25%	29.750000	43000.000000	0.000000
50%	37.000000	70000.000000	0.000000
75%	46.000000	88000.000000	1.000000
max	60.000000	150000.000000	1.000000

In [95]:

```
df.value_counts()
```

Out[95]:

```
Gender  Age  EstimatedSalary  Purchased
Female  41   72000            0            3
Male    40   57000            0            3
        42   65000            0            2
             54000            0            2
        35   75000            0            2
                                           ..
Female  42   90000            1            1
             80000            1            1
             79000            0            1
             75000            0            1
Male    60   102000           1            1
Length: 380, dtype: int64
```

In [96]:

```
df.dtypes
```

Out[96]:

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

In [97]:

```
df.isnull().sum()
```

Out[97]:

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

In [98]:

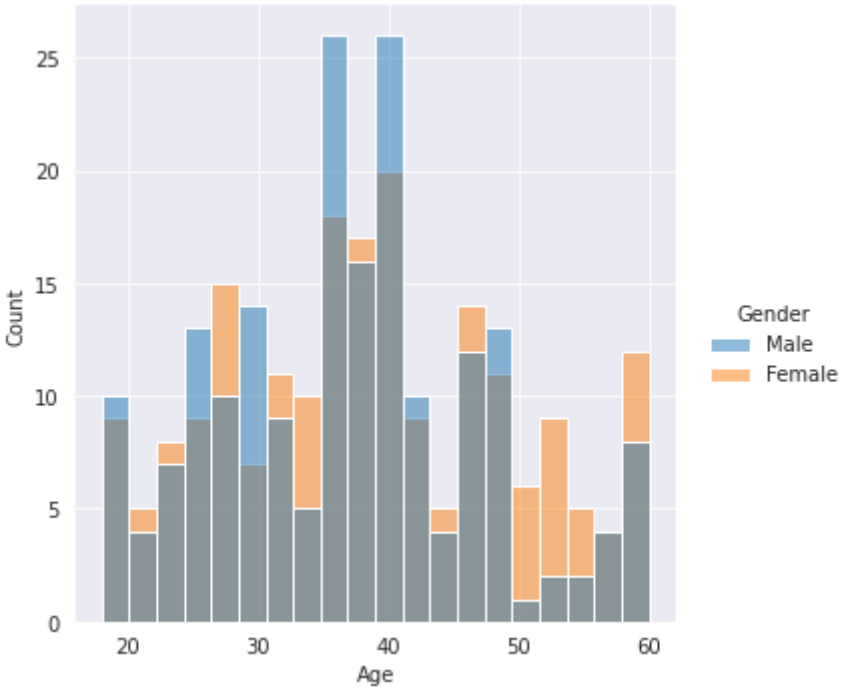
```
sns.set_style("darkgrid")
```

In [99]:

```
sns.displot(data=df, x="Age", bins=20, hue="Gender")
```

Out[99]:

```
<seaborn.axisgrid.FacetGrid at 0x7f8316451e50>
```



In [100]:

```
sns.scatterplot(data=df, x="Age", y="EstimatedSalary", hue="Purchased")
```

Out[100]:

```
<AxesSubplot:xlabel='Age', ylabel='EstimatedSalary'>
```

In [101]:

```
male = pd.get_dummies(df["Gender"], drop_first=True)
male
```

Out[101]:

	Male
0	1
1	1
2	0
3	0
4	1
...	...
395	0
396	1
397	0
398	1
399	0

400 rows × 1 columns

In [73]:

```
df.head()
```

Out[73]:

	Gender	Age	EstimatedSalary	Purchased
0	Male	19	19000	0
1	Male	35	20000	0
2	Female	26	43000	0
3	Female	27	57000	0
4	Male	19	76000	0

In [74]:

```
df = pd.concat([df, male], axis=1)
df.drop("Gender", axis=1, inplace=True)
```

In [75]:

```
df.head()
```

Out[75]:

	Age	EstimatedSalary	Purchased	Male
0	19	19000	0	1
1	35	20000	0	1
2	26	43000	0	0
3	27	57000	0	0
4	19	76000	0	1

In [76]:

```
X = df.drop("Purchased", axis=1)
y = df["Purchased"]
```

In [77]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=4
```

In [78]:

```
from sklearn.linear_model import LogisticRegression
```

In [79]:

```
model = LogisticRegression()
```

In [80]:

```
model.fit(X_train,y_train)
```

Out[80]:

```
LogisticRegression()
```

In [81]:

```
y_pred = model.predict(X_test)
```

In [82]:

```
from sklearn.metrics import confusion_matrix, classification_report
```

In [83]:

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

```
0.6060606060606061
```

In [84]:

```
print(confusion_matrix(y_test, y_pred))
```

```
[[80  0]
 [52  0]]
```

In [85]:

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.61      1.00      0.75        80
           1       0.00      0.00      0.00        52

    accuracy                           0.61       132
   macro avg       0.30      0.50      0.38       132
weighted avg       0.37      0.61      0.46       132
```

```
/usr/local/lib64/python3.8/site-packages/sklearn/metrics/_classification.p
y:1318: UndefinedMetricWarning: Precision and F-score are ill-defined and
being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib64/python3.8/site-packages/sklearn/metrics/_classification.p
y:1318: UndefinedMetricWarning: Precision and F-score are ill-defined and
being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib64/python3.8/site-packages/sklearn/metrics/_classification.p
y:1318: UndefinedMetricWarning: Precision and F-score are ill-defined and
being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [86]:

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

```
precision: [0.60606061 0.        ]
Recall: [1. 0.]
fscore: [0.75471698 0.        ]
support: [80 52]
```

```
/usr/local/lib64/python3.8/site-packages/sklearn/metrics/_classification.p
y:1318: UndefinedMetricWarning: Precision and F-score are ill-defined and
being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [ ]:

```

```

In [ ]:

```

```