In [5]:
```
%matplotlib inline
import matplotlib
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

In [6]:
```
dataset=pd.read_csv("suv_data.csv")
```

In [7]:
```
dataset.head(10)
```

Out[7]:

	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
5	15728773	Male	27	58000	0
6	15598044	Female	27	84000	0
7	15694829	Female	32	150000	1
8	15600575	Male	25	33000	0
9	15727311	Female	35	65000	0

Train

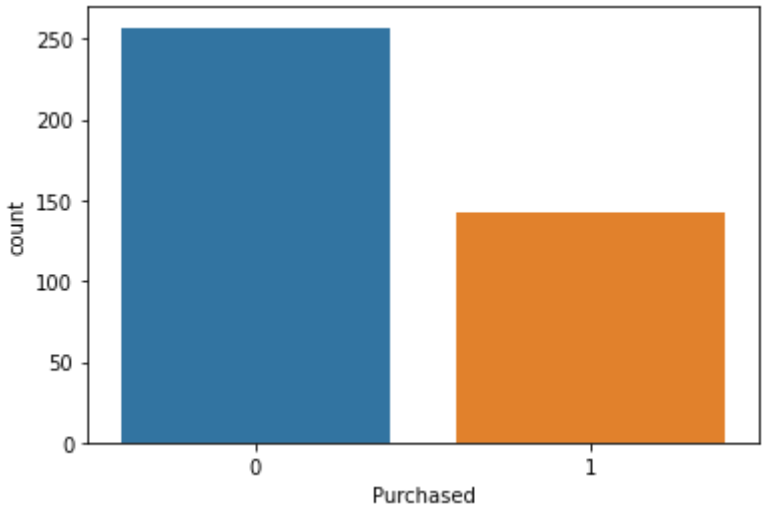
In [8]:
```
print(len(dataset.index))
```

```
400
```

```
#Analyzing
```

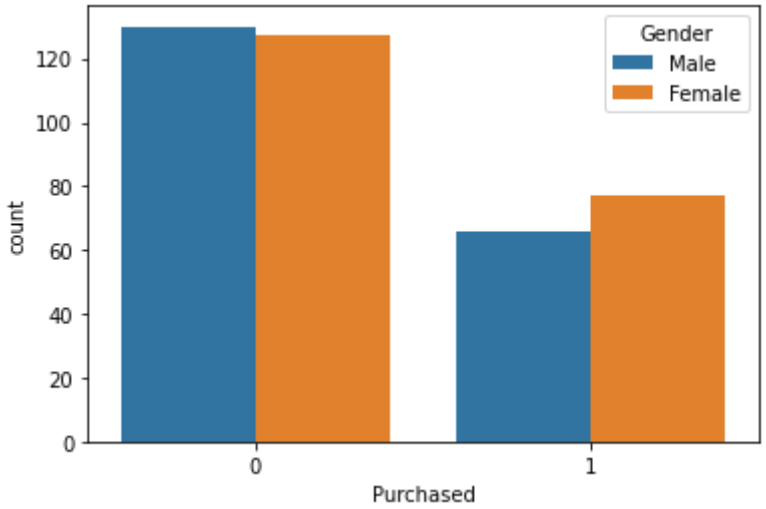
In [9]:
```
sns.countplot(x="Purchased", data=dataset)
```

Out[9]: <AxesSubplot:xlabel='Purchased', ylabel='count'>



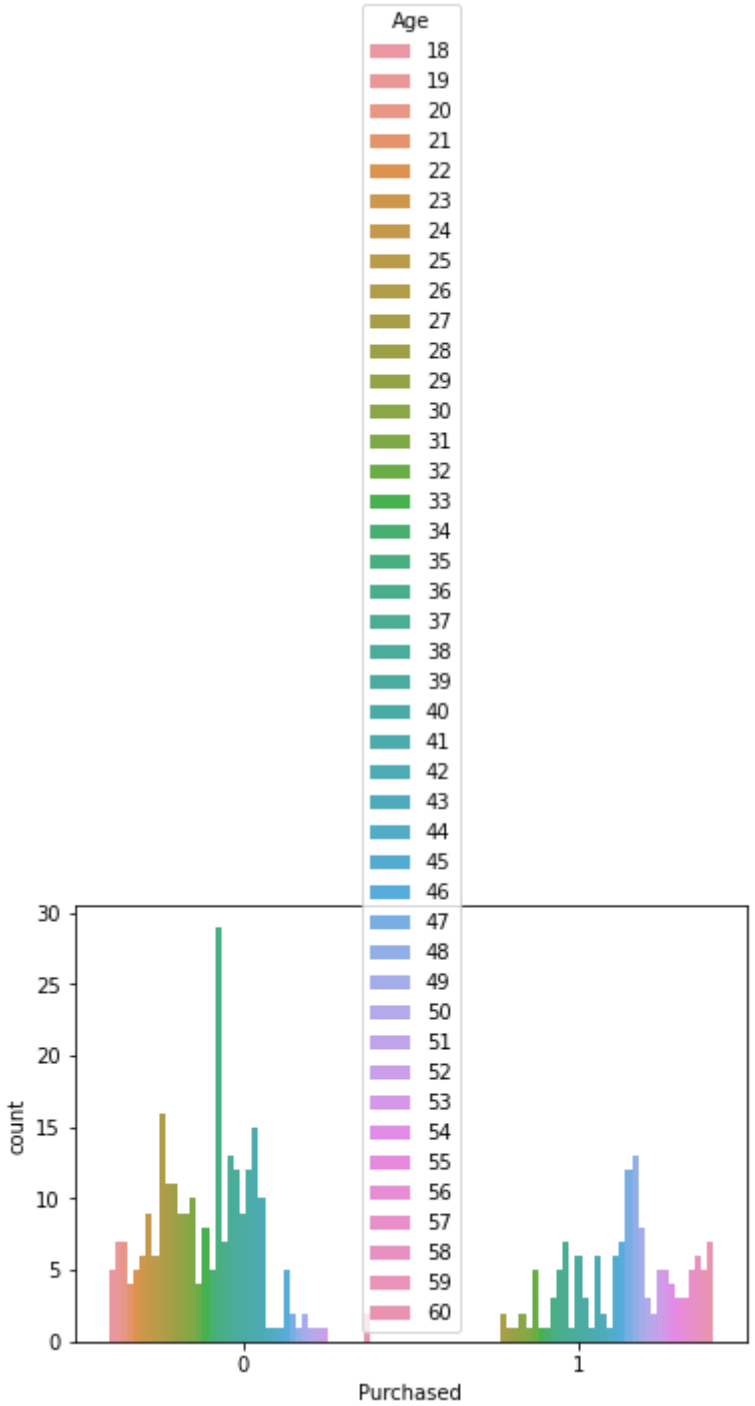
In [10]:
```
sns.countplot(x="Purchased", hue="Gender", data=dataset)
```

Out[10]: <AxesSubplot:xlabel='Purchased', ylabel='count'>



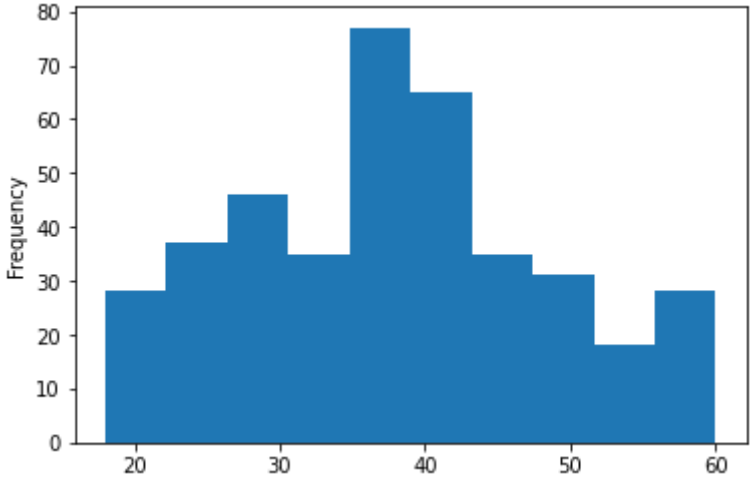
In [15]:
```
sns.countplot(x="Purchased", hue="Age" ,data=dataset)
```

Out[15]:
```
<AxesSubplot:xlabel='Purchased', ylabel='count'>
```


Age
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
30
25
20
15
10
5
0
count
0
1
Purchased

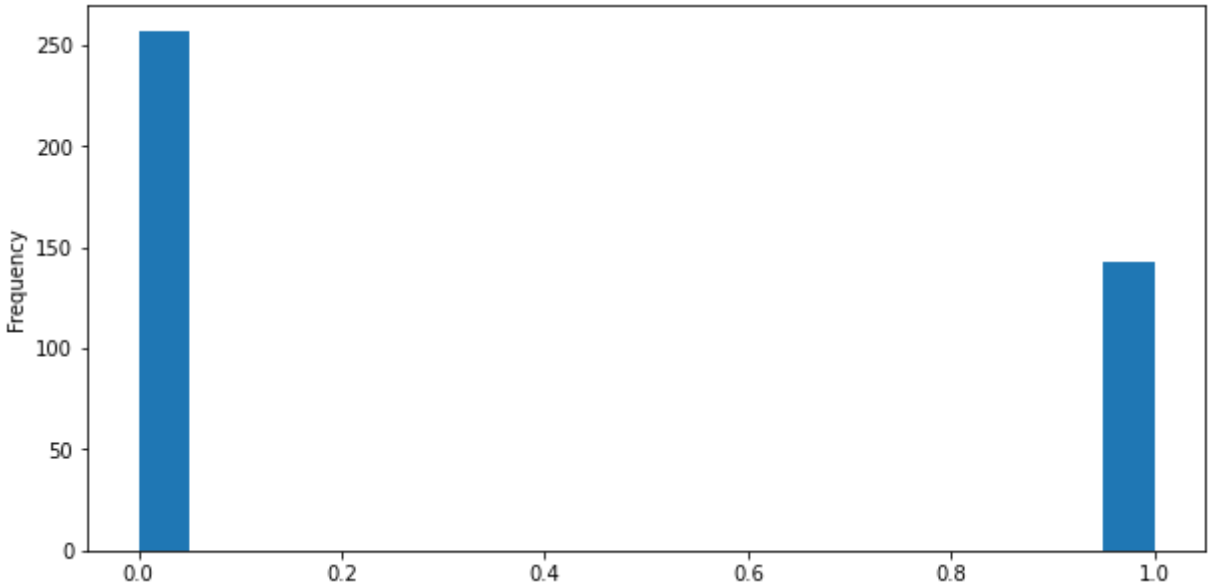
In [16]:
```
dataset["Age"].plot.hist()
```

Out[16]:
```
<AxesSubplot:ylabel='Frequency'>
```



In [18]:
```
dataset["Purchased"].plot.hist(bins=20, figsize=(10,5))
```

Out[18]:
```
<AxesSubplot:ylabel='Frequency'>
```



In [19]:
```
dataset.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
```

Data Wrangling

In [20]:
```
dataset.isnull()
```

Out[20]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
395	False	False	False	False	False
396	False	False	False	False	False
397	False	False	False	False	False
398	False	False	False	False	False
399	False	False	False	False	False

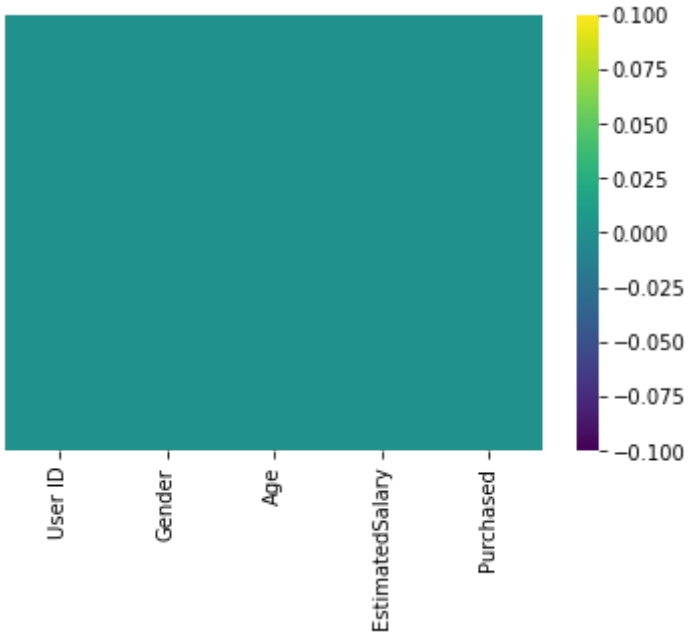
400 rows × 5 columns

In [21]:
```
dataset.isnull().sum()
```

Out[21]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

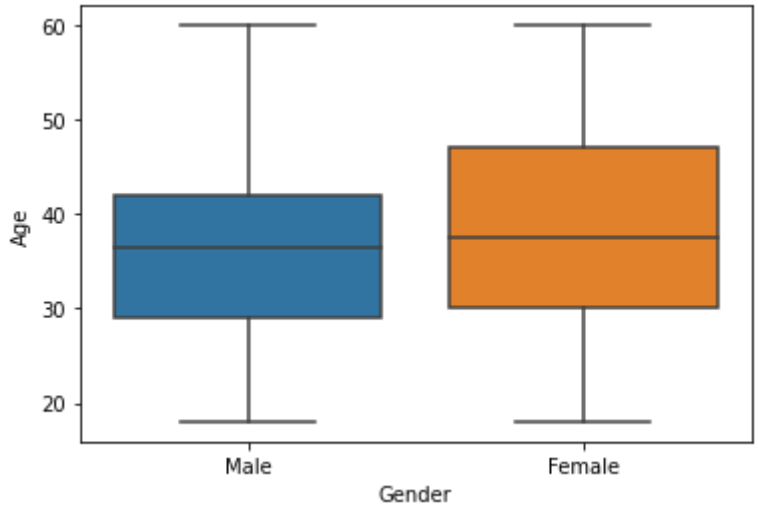
In [22]:
```
sns.heatmap(dataset.isnull(), yticklabels=False, cmap="viridis")
```

Out[22]: <AxesSubplot:>



In [23]:
```
sns.boxplot(x="Gender", y="Age", data=dataset)
```

Out[23]:
```
<AxesSubplot:xlabel='Gender', ylabel='Age'>
```



In [24]:
```
dataset.head(5)
```

Out[24]:

	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 [25]:
```
sex=pd.get_dummies(dataset['Gender'],drop_first=True)
sex.head(5)
```

Out[25]:

	Male
0	1
1	1
2	0
3	0
4	1

In [26]:
```
Purchased=pd.get_dummies(dataset['Purchased'],drop_first=True)
Purchased.head(5)
```

Out[26]:

	1
0	0
1	0
2	0
3	0
4	0

In [28]:
```
dataset=pd.concat([dataset,sex,Purchased],axis=1)
```

In [29]:
```
dataset.head(5)
```

Out[29]:

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

In [30]:
```
dataset.drop(['Gender','User ID'],axis=1,inplace=True)
```

In [31]:
```
dataset.head(5)
```

Out[31]:

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

```
Train/Test Data
```

In [ ]:
```
X=dataset.iloc[:,[2,3]].values   #independent variable (in this age and salary)
y=dataset.iloc[:,4].values      #dependent variable   (in this purchased)
```

In [38]:
```
y
```

Out[38]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1,
       1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1,
       0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 0, 1], dtype=int64)
```

In [39]:
```
from sklearn.model_selection import train_test_split
```

In [40]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```

In [41]:
```
from sklearn.preprocessing import StandardScaler
```

In [42]:
```
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

In [43]:
```
from sklearn.linear_model import LogisticRegression
```

In [56]:
```
clf= LogisticRegression(random_state=0).fit(X_train,y_train)
```

In [57]:
```
y_pred=clf.predict(X_test)
```

In [58]:
```
from sklearn.metrics import accuracy_score
```

In [60]:
```
accuracy_score(y_test,y_pred)*100
```

Out[60]: 89.0

In [ ]:
```

```