Importing the Dependencies

In [49]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
```

Data Collection and Processing

In [9]:

```
#loading the dataset into pandas DataFrame
loan_dataset=pd.read_csv("C:\\Users\\manan\\OneDrive\\Desktop\\train_u6lujuX_CVtuZ9i (1).csv")
```

In [10]:

```
loan_dataset
```

Out[10]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	
...	...	...	...	...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	

614 rows × 13 columns

In [11]:

```
loan_dataset.head(5)
```

Out[11]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_H
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [13]:

```
#number of rows and coloumns
loan_dataset.shape
```

```
Out[13]: (614, 13)
```

```
In [14]: #statistical measures
         loan_dataset.describe()
```

Out[14]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

```
In [16]: #numbe of missing values in each coloumn
         loan_dataset.isnull().sum()
```

```
Out[16]: Loan_ID               0
         Gender               13
         Married               3
         Dependents           15
         Education             0
         Self_Employed        32
         ApplicantIncome       0
         CoapplicantIncome     0
         LoanAmount           22
         Loan_Amount_Term     14
         Credit_History       50
         Property_Area         0
         Loan_Status           0
         dtype: int64
```

```
In [17]: #dropping the missing values
         loan_dataset=loan_dataset.dropna()
```

```
In [18]: #numbe of missing values in each coloumn
         loan_dataset.isnull().sum()
```

Out[18]:

```
Loan_ID              0
Gender               0
Married              0
Dependents           0
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
dtype: int64
```

In [19]:

```
#label encoding
loan_dataset.replace({"Loan_Status":{'N':0,'Y':1}},inplace=True)
```

```
C:\Users\manan\AppData\Local\Temp\ipykernel_18320\3097104041.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  loan_dataset.replace({"Loan_Status":{'N':0,'Y':1}},inplace=True)
```

In [20]:

```
loan_dataset
```

Out[20]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	
5	LP001011	Male	Yes	2	Graduate	Yes	5417	4196.0	267.0	360.0	
...	...	...	...	...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	

480 rows × 13 columns

In [22]:

```
loan_dataset['Dependents'].value_counts()
```

Out[22]:

```
Dependents
0     274
2      85
1      80
3+     41
Name: count, dtype: int64
```

In [24]:

```
#we need to replace 3+ values to 4
loan_dataset = loan_dataset.replace(to_replace='3+', value='4')
```

In [25]:

```
loan_dataset['Dependents'].value_counts()
```

Out[25]:

```
Dependents
0    274
2     85
1     80
4     41
Name: count, dtype: int64
```

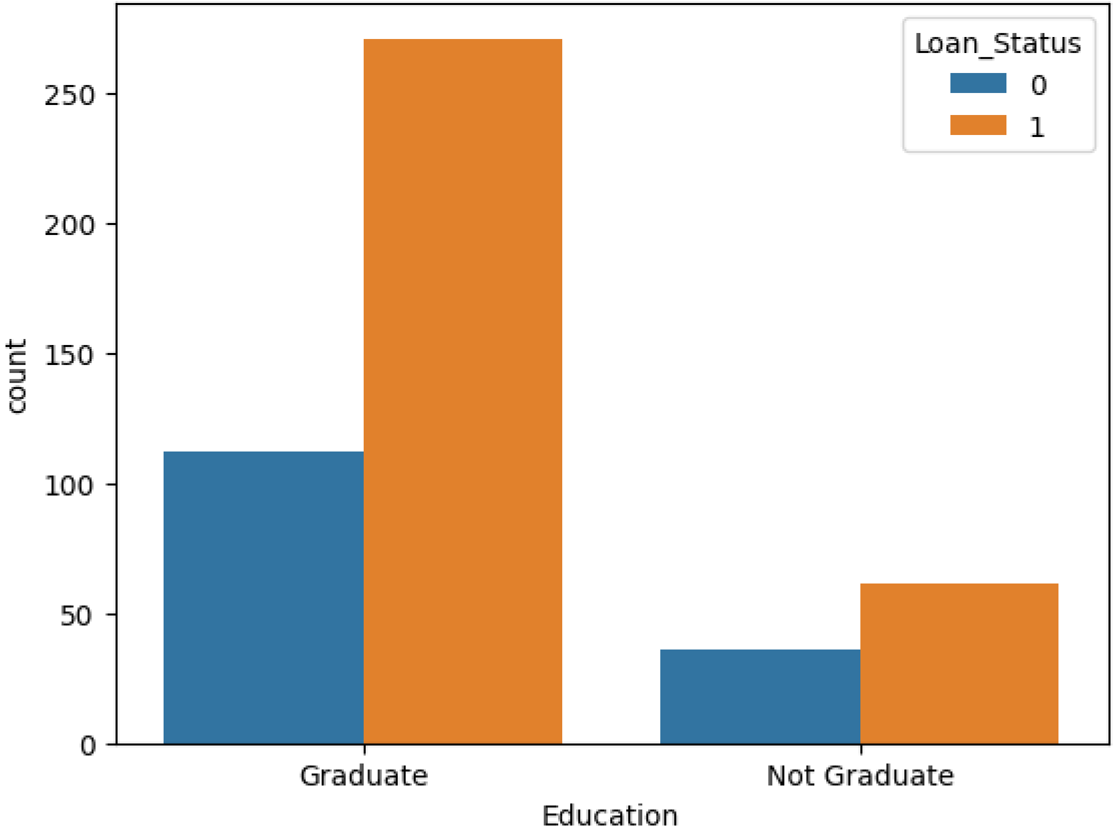
Data Visualization

In [30]:

```
#education and loan status
sns.countplot(x="Education",hue="Loan_Status",data=loan_dataset)
```

Out[30]:

```
<Axes: xlabel='Education', ylabel='count'>
```

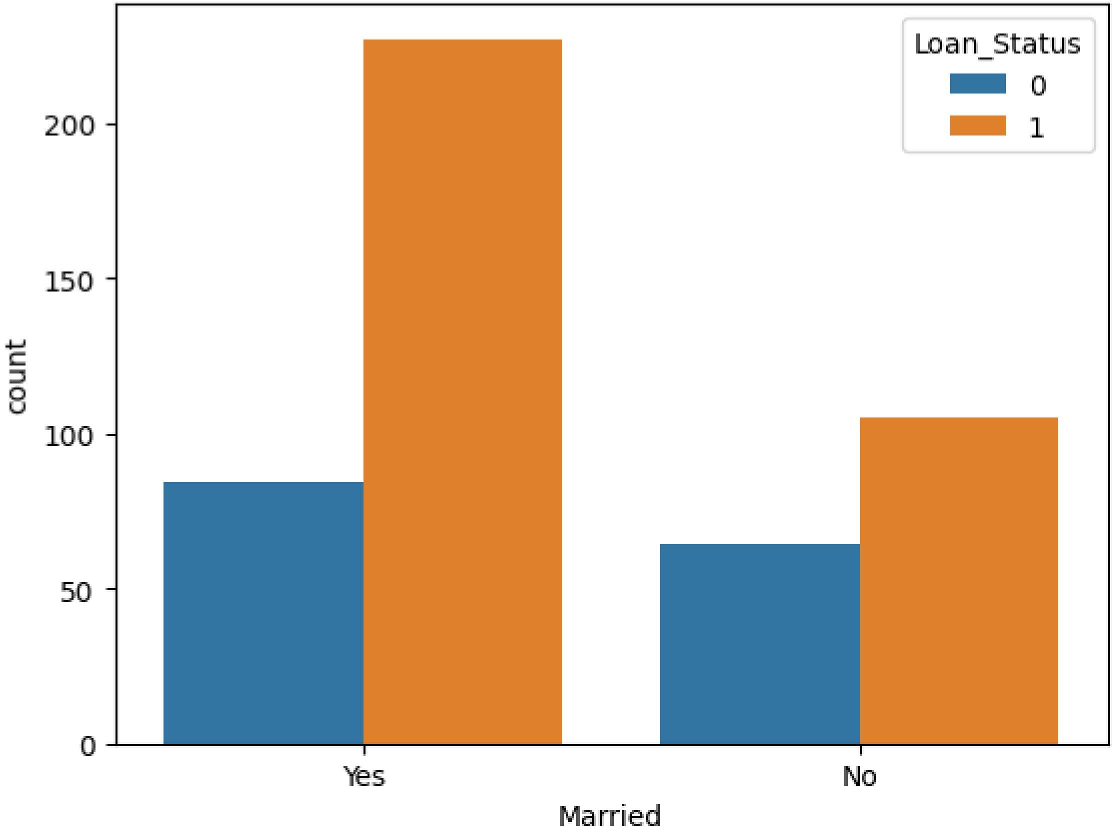


In [32]:

```
#marital status & loan status
sns.countplot(x="Married",hue="Loan_Status",data=loan_dataset)
```

Out[32]:

```
<Axes: xlabel='Married', ylabel='count'>
```



Converting Categorical Data into Numerical Values

In [36]:

```python
loan_dataset.replace({'Married':{'No':0,'Yes':1},'Gender':{'Male':1,'Female':0},'Self_Employed':{'No':0,'Yes':1},
                      'Property_Area':{'Rural':0,'Semiurban':1,'Urban':2},'Education':{'Graduate':1,'Not Graduate':0}},inplace=True)
```

In [37]:

```python
loan_dataset.head()
```

Out[37]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_H
1	LP001003	1	1	1	1	0	4583	1508.0	128.0	360.0	
2	LP001005	1	1	0	1	1	3000	0.0	66.0	360.0	
3	LP001006	1	1	0	0	0	2583	2358.0	120.0	360.0	
4	LP001008	1	0	0	1	0	6000	0.0	141.0	360.0	
5	LP001011	1	1	2	1	1	5417	4196.0	267.0	360.0	

Seperating the Data and Label

In [40]:

```
X=loan_dataset.drop(columns=['Loan_ID','Loan_Status'],axis=1)
Y=loan_dataset['Loan_Status']
```

In [41]:

```
loan_dataset.head()
```

Out[41]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_H
1	LP001003	1	1	1	1	0	4583	1508.0	128.0	360.0	
2	LP001005	1	1	0	1	1	3000	0.0	66.0	360.0	
3	LP001006	1	1	0	0	0	2583	2358.0	120.0	360.0	
4	LP001008	1	0	0	1	0	6000	0.0	141.0	360.0	
5	LP001011	1	1	2	1	1	5417	4196.0	267.0	360.0	

In [43]:

```
print(X)
print(Y)
```

```
     Gender  Married Dependents  Education  Self_Employed  ApplicantIncome  \
1         1        1          1          1              0             4583   
2         1        1          0          1              1             3000   
3         1        1          0          0              0             2583   
4         1        0          0          1              0             6000   
5         1        1          2          1              1             5417   
..      ...      ...        ...        ...            ...              ...   
609       0        0          0          1              0             2900   
610       1        1          4          1              0             4106   
611       1        1          1          1              0             8072   
612       1        1          2          1              0             7583   
613       0        0          0          1              1             4583   

     CoapplicantIncome  LoanAmount  Loan_Amount_Term  Credit_History  \
1               1508.0       128.0             360.0             1.0   
2                  0.0        66.0             360.0             1.0   
3               2358.0       120.0             360.0             1.0   
4                  0.0       141.0             360.0             1.0   
5               4196.0       267.0             360.0             1.0   
..                 ...         ...               ...             ...   
609                0.0        71.0             360.0             1.0   
610                0.0        40.0             180.0             1.0   
611              240.0       253.0             360.0             1.0   
612                0.0       187.0             360.0             1.0   
613                0.0       133.0             360.0             0.0   

     Property_Area  
1                0  
2                2  
3                2  
4                2  
5                2  
..             ...  
609              0  
610              0  
611              2  
612              2  
613              1  

[480 rows x 11 columns]
1      0
2      1
3      1
4      1
5      1
      ..
```

```
609    1
610    1
611    1
612    1
613    0
Name: Loan_Status, Length: 480, dtype: int64
```

Training and Test Data Seperation

In [50]:
```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.1,stratify=Y,random_state=2)
```

In [51]:
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(480, 11) (432, 11) (48, 11)
```

TRAINING THE MODEL USING SVM: SUPPORT VECTOR MACHINE

In [52]:
```
classifier = svm.SVC(kernel='linear')
```

In [53]:
```
#training the support vector machine
```

In [54]:
```
classifier.fit(X_train,Y_train)
```

Out[54]:

```
▾ SVC
SVC(kernel='linear')
```

Model Evaluation

In [56]:
```
#accuracy score
```

In [57]:
```
X_train_predicition= classifier.predict(X_train)
training_data_accuracy=accuracy_score(X_train_predicition,Y_train)
```

In [58]:
```
X_train_predicition
training_data_accuracy
```

Out[58]:
```
0.7986111111111112
```

In [59]:
```
X_test_predicition= classifier.predict(X_test)
test_data_accuracy=accuracy_score(X_test_predicition,Y_test)
```

```
In [60]: X_test_predicition
         test_data_accuracy
```

```
Out[60]: 0.8333333333333334
```

Making a Predctive System

```
In [ ]:
```