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

```
loan_data=pd.read_csv('/content/prediction.csv')
```

```
loan_data=loan_data.dropna()
```

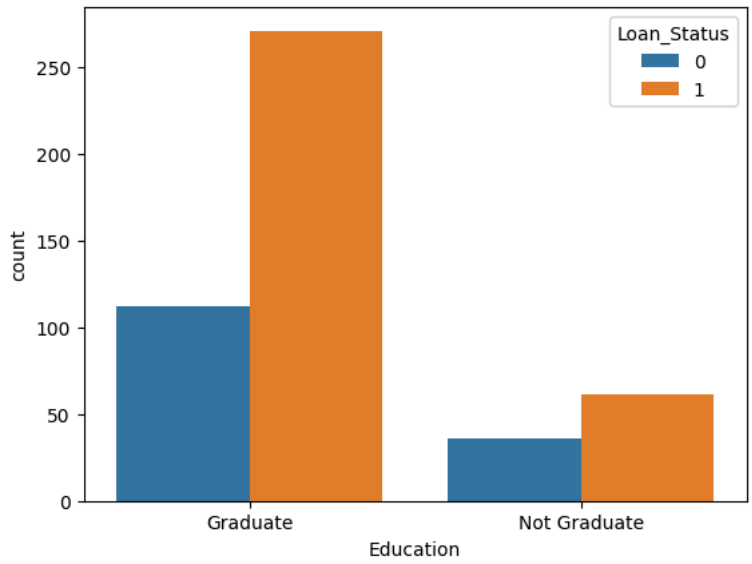
```
loan_data.replace({"Loan_Status": {'N': 0, "Y": 1}}, inplace=True)
```

```
/tmp/ipython-input-2603371208.py:1: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in a
  loan_data.replace({"Loan_Status": {'N': 0, "Y": 1}}, inplace=True)
```

```
loan_data=loan_data.replace(to_replace='3+',value=4)
```

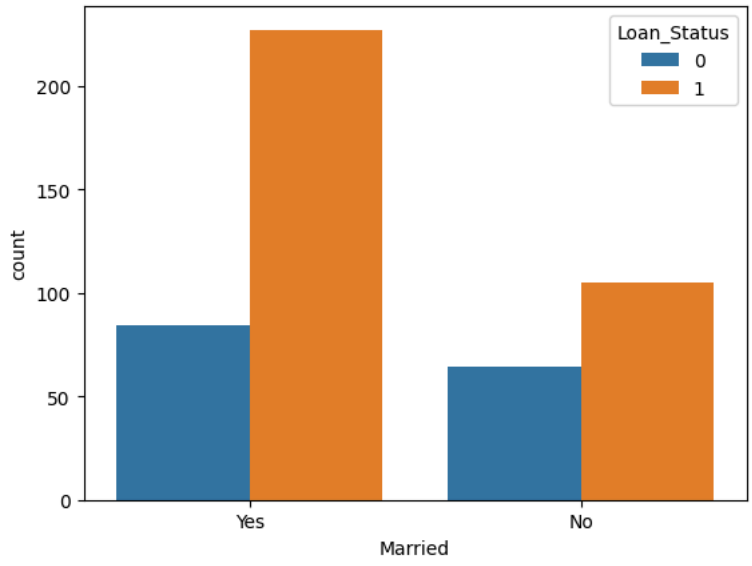
```
sns.countplot(x='Education',hue='Loan_Status',data=loan_data)
```

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



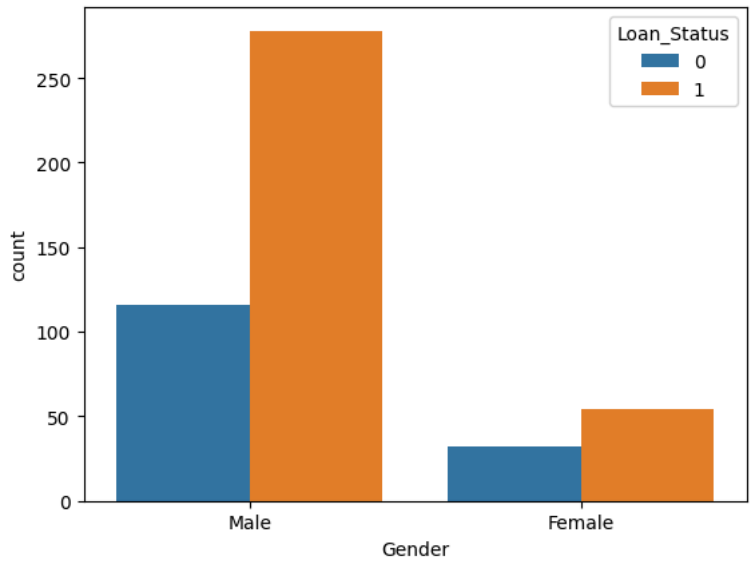

```
sns.countplot(x='Married',hue='Loan_Status',data=loan_data)
```

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




```
sns.countplot(x='Gender',hue='Loan_Status',data=loan_data)
```

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




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

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

```
X_train,X_test,Y_train,Y_test=train_test_split(X,y,test_size=0.1,stratify=y,random_state=2)
```

```
print(X.shape,X_train.shape,X_test.shape)
```

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

Train the model using supprt vector machine

```
classifier=svm.SVC(kernel='linear')
```

Training the support vector model

```
loan_data.describe()
```

	Gender	Married	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	C
count	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	
mean	0.820833	0.647917	0.797917	0.137500	5364.231250	1581.093583	144.735417	342.050000	
std	0.383892	0.478118	0.401973	0.344734	5668.251251	2617.692267	80.508164	65.212401	
min	0.000000	0.000000	0.000000	0.000000	150.000000	0.000000	9.000000	36.000000	
25%	1.000000	0.000000	1.000000	0.000000	2898.750000	0.000000	100.000000	360.000000	
50%	1.000000	1.000000	1.000000	0.000000	3859.000000	1084.500000	128.000000	360.000000	
75%	1.000000	1.000000	1.000000	0.000000	5852.500000	2253.250000	170.000000	360.000000	
max	1.000000	1.000000	1.000000	1.000000	81000.000000	33837.000000	600.000000	480.000000	

```
loan_data.shape
```

```
(480, 13)
```

```
loan_data.head
```

```
pandas.core.generic.NDFrame.head
def head(n: int=5) -> Self
```

/usr/local/lib/python3.12/dist-packages/pandas/core/generic.py
Return the first `n` rows.

This function returns the first `n` rows for the object based
on position. It is useful for quickly testing if your object
has the right type of data in it.

```
loan_data.head(10)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Am
1	LP001003	1	1	1	1	0	4583	1508.0	128.0	
2	LP001005	1	1	0	1	1	3000	0.0	66.0	
3	LP001006	1	1	0	0	0	2583	2358.0	120.0	
4	LP001008	1	0	0	1	0	6000	0.0	141.0	
5	LP001011	1	1	2	1	1	5417	4196.0	267.0	
6	LP001013	1	1	0	0	0	2333	1516.0	95.0	
7	LP001014	1	1	4	1	0	3036	2504.0	158.0	
8	LP001018	1	1	2	1	0	4006	1526.0	168.0	
9	LP001020	1	1	1	1	0	12841	10968.0	349.0	
10	LP001024	1	1	2	1	0	3200	700.0	70.0	

Next steps: Generate code with loan_data | New interactive sheet

```
# Separate features & target
X = loan_data.drop('Loan_Status', axis=1)
Y = loan_data['Loan_Status'].replace({'Y':1, 'N':0})  # convert target to 1/0

# Convert ALL categorical columns to numeric
X = pd.get_dummies(X, drop_first=True)
```

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

```
X_train,X_test,Y_train,Y_test=train_test_split(X,y,test_size=0.1,stratify=y,random_state=2)
```

```
classifier=svm.SVC(kernel='linear')
```

```
classifier.fit(X_train,Y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipython-input-2330531615.py in <cell line: 0>()
----> 1 classifier.fit(X_train,Y_train)

                                 6 frames
/usr/local/lib/python3.12/dist-packages/pandas/core/generic.py in __array__(self, dtype, copy)
   2151     ) -> np.ndarray:
   2152         values = self._values
-> 2153         arr = np.asarray(values, dtype=dtype)
   2154         if (
   2155             astype_is_view(values.dtype, arr.dtype)

ValueError: could not convert string to float: 'Semiurban'
```

Next steps: Explain error

```
X_train.dtypes
```

	0
Gender	int64
Married	int64
Dependents	object
Education	int64
Self_Employed	int64
ApplicantIncome	int64
CoapplicantIncome	float64
LoanAmount	float64
Loan_Amount_Term	float64

```
loan_data['Dependents'] = loan_data['Dependents'].replace({'3+': 3}).astype(int)
```

```
loan_data['Property_Area'] = loan_data['Property_Area'].map({
    'Rural': 0,
    'Semiurban': 1,
    'Urban': 2
})
```

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

```
X_train,X_test,Y_train,Y_test=train_test_split(X,y,test_size=0.1,stratify=y,random_state=2)
```

```
classifier=svm.SVC(kernel='linear')
```

```
classifier.fit(X_train,Y_train)
```

```
SVC
SVC(kernel='linear')
```

```
x_train_prediction=classifier.predict(X_train)
training_data_accuracy=accuracy_score(x_train_prediction,Y_train)
```

```
print(training_data_accuracy)
```

```
0.7986111111111112
```

```
x_test_prediction=classifier.predict(X_test)
test_data_accuracy=accuracy_score(x_test_prediction,Y_test)
```

```
print(test_data_accuracy)
```

```
0.8333333333333334
```