In [51]:
```
#Importing the Libraries
import numpy as np
import pandas as pd
import pickle
import matplotlib.pyplot as plt
#%matplotlib.pylot as plot
import seaborn as sns
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier,RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
import imblearn
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,f1_s
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [51], in <cell line: 13>()
     11 from sklearn.neighbors import KNeighborsClassifier
     12 from sklearn.model_selection import RandomizedSearchCV
---> 13 import imblearn
     14 from sklearn.model_selection import train_test_split
     15 from sklearn.preprocessing import StandardScaler

ModuleNotFoundError: No module named 'imblearn'
```

In [36]:
```
#Data Collection and Preparation
#Read The Data Set
df=pd.read_csv("E:\\NMDS\pl_train.csv")
df.head()
```

Out[36]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantI
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	

In [37]:
```
df.tail()
```

Out[37]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

In [38]:
```
#Data Collection and Preparation
#Read The Data Set
df1=pd.read_csv("E:\\NMDS\pl_test.csv")
df1.head()
```

Out[38]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantI
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	

In [39]:
```
df1.tail()
```

Out[39]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

In [40]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Loan_ID            614 non-null    object 
 1   Gender             601 non-null    object 
 2   Married            611 non-null    object 
 3   Dependents         599 non-null    object 
 4   Education          614 non-null    object 
 5   Self_Employed      582 non-null    object 
 6   ApplicantIncome    614 non-null    int64  
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object 
 12  Loan_Status        614 non-null    object 
dtypes: float64(4), int64(1), object(8)
memory usage: 43.2+ KB
```

In [41]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 367 entries, 0 to 366
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Loan_ID            367 non-null    object 
 1   Gender             356 non-null    object 
 2   Married            367 non-null    object 
 3   Dependents         357 non-null    object 
 4   Education          367 non-null    object 
 5   Self_Employed      344 non-null    object 
 6   ApplicantIncome    367 non-null    int64  
 7   CoapplicantIncome  367 non-null    int64  
 8   LoanAmount         362 non-null    float64
 9   Loan_Amount_Term   361 non-null    float64
 10  Credit_History     338 non-null    float64
 11  Property_Area      367 non-null    object 
dtypes: float64(3), int64(2), object(7)
memory usage: 24.4+ KB
```

In [42]:
```
df.isnull().sum()
```

Out[42]:
```
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 [43]:
```
df1.isnull().sum()
```

Out[43]:
```
Loan_ID               0
Gender               11
Married               0
Dependents           10
Education             0
Self_Employed        23
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount            5
Loan_Amount_Term      6
Credit_History       29
Property_Area         0
dtype: int64
```

In [44]:
```
df['Gender']=df['Gender'].fillna(df['Gender'].mode()[0])
df['Married']=df['Married'].fillna(df['Married'].mode()[0])
```

In [45]:
```
df1['Gender']=df1['Gender'].fillna(df1['Gender'].mode()[0])
df1['Married']=df1['Married'].fillna(df1['Married'].mode()[0])
```

In [46]:
```
#replacing + with space for filling the nan values
df['Dependents']=df['Dependents'].str.replace('+','')
df['Dependents']=df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed']=df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['LoanAmount']=df['LoanAmount'].fillna(df['LoanAmount'].mode()[0])
df['Loan_Amount_Term']=df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0]
df['Credit_History']=df['Credit_History'].fillna(df['Credit_History'].mode()[0])
```

```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_4764\1052807915.py:2: FutureWarni
ng: The default value of regex will change from True to False in a future version. In
addition, single character regular expressions will *not* be treated as literal strin
gs when regex=True.
  df['Dependents']=df['Dependents'].str.replace('+','')
```

In [47]:
```
#replacing + with space for filling the nan values
df1['Dependents']=df1['Dependents'].str.replace('+','')
df1['Dependents']=df1['Dependents'].fillna(df1['Dependents'].mode()[0])
df1['Self_Employed']=df1['Self_Employed'].fillna(df1['Self_Employed'].mode()[0])
df1['LoanAmount']=df1['LoanAmount'].fillna(df1['LoanAmount'].mode()[0])
df1['Loan_Amount_Term']=df1['Loan_Amount_Term'].fillna(df1['Loan_Amount_Term'].mode()
df1['Credit_History']=df1['Credit_History'].fillna(df1['Credit_History'].mode()[0])
```

```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_4764\1127294072.py:2: FutureWarni
ng: The default value of regex will change from True to False in a future version. In
addition, single character regular expressions will *not* be treated as literal strin
gs when regex=True.
  df1['Dependents']=df1['Dependents'].str.replace('+','')
```

In [48]:
```
#changing the data type of each float column to int
from numpy import int64
df['Gender']=df['Gender'].astype('int64')
df['Married']=df['Married'].astype(int64)
df['Dependents']=df['Dependents'].astype(int64)
df['Dependents']=df['Dependents'].astype(int64)
df['Self_Employed']=df['Self_Employed'].astype(int64)
df['CoapplicantIncome']=df['CoapplicantIncome'].astype(int64)
df['LoanAmount']=df['LoanAmount'].astype(int64)
```

```
df['Loan_Amount_Term']=df['Loan_Amount_Term'].astype(int64)
df['Credit_History']=df['Credit_History'].astype(int64)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [48], in <cell line: 3>()
      1 #changing the data type of each float column to int
      2 from numpy import int64
----> 3 df['Gender']=df['Gender'].astype('int64')
      4 df['Married']=df['Married'].astype(int64)
      5 df['Dependents']=df['Dependents'].astype(int64)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py:5912, in NDFra
me.astype(self, dtype, copy, errors)
   5905     results = [
   5906         self.iloc[:, i].astype(dtype, copy=copy)
   5907         for i in range(len(self.columns))
   5908     ]
   5910 else:
   5911     # else, only a single dtype is given
-> 5912     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   5913     return self._constructor(new_data).__finalize__(self, method="astype")
   5915 # GH 33113: handle empty frame or series

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\managers.py:41
9, in BaseBlockManager.astype(self, dtype, copy, errors)
    418 def astype(self: T, dtype, copy: bool = False, errors: str = "raise") -> T:
--> 419     return self.apply("astype", dtype=dtype, copy=copy, errors=errors)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\managers.py:30
4, in BaseBlockManager.apply(self, f, align_keys, ignore_failures, **kwargs)
    302         applied = b.apply(f, **kwargs)
    303     else:
--> 304         applied = getattr(b, f)(**kwargs)
    305 except (TypeError, NotImplementedError):
    306     if not ignore_failures:

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\blocks.py:580,
in Block.astype(self, dtype, copy, errors)
    562 """
    563 Coerce to the new dtype.
    564 
   (...)
    576 Block
    577 """
    578 values = self.values
--> 580 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    582 new_values = maybe_coerce_values(new_values)
    583 newb = self.make_block(new_values)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1292, in a
stype_array_safe(values, dtype, copy, errors)
   1289     dtype = dtype.numpy_dtype
   1291 try:
-> 1292     new_values = astype_array(values, dtype, copy=copy)
   1293 except (ValueError, TypeError):
   1294     # e.g. astype_nansafe can fail on object-dtype of strings
   1295     #  trying to convert to float
   1296     if errors == "ignore":

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1237, in a
stype_array(values, dtype, copy)
   1234     values = values.astype(dtype, copy=copy)
   1236 else:
-> 1237     values = astype_nansafe(values, dtype, copy=copy)
   1239 # in pandas we don't store numpy str dtypes, so convert to object
   1240 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1154, in a
stype_nansafe(arr, dtype, copy, skipna)
   1150 elif is_object_dtype(arr.dtype):
   1151
   1152     # work around NumPy brokenness, #1987
   1153     if np.issubdtype(dtype.type, np.integer):
-> 1154         return lib.astype_intsafe(arr, dtype)
   1156     # if we have a datetime/timedelta array of objects
   1157     # then coerce to a proper dtype and recall astype_nansafe
   1159     elif is_datetime64_dtype(dtype):

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\_libs\lib.pyx:668, in pandas._
libs.lib.astype_intsafe()

ValueError: invalid literal for int() with base 10: 'Male'
```

In [49]:

```
#changing the data type of each float column to int
from numpy import int64
df1['Gender']=df1['Gender'].astype('int64')
df1['Married']=df1['Married'].astype(int64)
df1['Dependents']=df1['Dependents'].astype(int64)
df1['Dependents']=df1['Dependents'].astype(int64)
df1['Self_Employed']=df1['Self_Employed'].astype(int64)
df1['CoapplicantIncome']=df1['CoapplicantIncome'].astype(int64)
df1['LoanAmount']=df1['LoanAmount'].astype(int64)
df1['Loan_Amount_Term']=df1['Loan_Amount_Term'].astype(int64)
df1['Credit_History']=df1['Credit_History'].astype(int64)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [49], in <cell line: 3>()
      1 #changing the data type of each float column to int
      2 from numpy import int64
----> 3 df1['Gender']=df1['Gender'].astype('int64')
      4 df1['Married']=df1['Married'].astype(int64)
      5 df1['Dependents']=df1['Dependents'].astype(int64)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py:5912, in NDFra
me.astype(self, dtype, copy, errors)
   5905     results = [
   5906         self.iloc[:, i].astype(dtype, copy=copy)
   5907         for i in range(len(self.columns))
   5908     ]
   5910 else:
   5911     # else, only a single dtype is given
-> 5912     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   5913     return self._constructor(new_data).__finalize__(self, method="astype")
   5915 # GH 33113: handle empty frame or series

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\managers.py:41
9, in BaseBlockManager.astype(self, dtype, copy, errors)
    418 def astype(self: T, dtype, copy: bool = False, errors: str = "raise") -> T:
--> 419     return self.apply("astype", dtype=dtype, copy=copy, errors=errors)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\managers.py:30
4, in BaseBlockManager.apply(self, f, align_keys, ignore_failures, **kwargs)
    302         applied = b.apply(f, **kwargs)
    303     else:
--> 304         applied = getattr(b, f)(**kwargs)
    305 except (TypeError, NotImplementedError):
    306     if not ignore_failures:

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\blocks.py:580,
in Block.astype(self, dtype, copy, errors)
    562 """
    563 Coerce to the new dtype.
    564 
   (...)
    576 Block
    577 """
    578 values = self.values
--> 580 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    582 new_values = maybe_coerce_values(new_values)
    583 newb = self.make_block(new_values)

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1292, in a
stype_array_safe(values, dtype, copy, errors)
   1289     dtype = dtype.numpy_dtype
   1291 try:
-> 1292     new_values = astype_array(values, dtype, copy=copy)
   1293 except (ValueError, TypeError):
   1294     # e.g. astype_nansafe can fail on object-dtype of strings
   1295     #  trying to convert to float
   1296     if errors == "ignore":

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1237, in a
stype_array(values, dtype, copy)
   1234     values = values.astype(dtype, copy=copy)
   1236 else:
-> 1237     values = astype_nansafe(values, dtype, copy=copy)
   1239 # in pandas we don't store numpy str dtypes, so convert to object
   1240 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py:1154, in a
stype_nansafe(arr, dtype, copy, skipna)
   1150 elif is_object_dtype(arr.dtype):
   1151 
   1152     # work around NumPy brokenness, #1987
   1153     if np.issubdtype(dtype.type, np.integer):
-> 1154         return lib.astype_intsafe(arr, dtype)
   1156     # if we have a datetime/timedelta array of objects
   1157     # then coerce to a proper dtype and recall astype_nansafe
   1159     elif is_datetime64_dtype(dtype):

File C:\ProgramData\Anaconda3\lib\site-packages\pandas\_libs\lib.pyx:668, in pandas._
libs.lib.astype_intsafe()

ValueError: invalid literal for int() with base 10: 'Male'
```

In [50]:
```
#Balancing the dataset by using smote
from imbalance.combine import SMOTETomek
smote=SMOTETomek(0.90)
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [50], in <cell line: 2>()
      1 #Balancing the dataset by using smote
----> 2 from imbalance.combine import SMOTETomek
      3 smote=SMOTETomek(0.90)

ModuleNotFoundError: No module named 'imbalance'
```

In [56]:
```
#dividing the dataset into dependent and independent y and x respectively
from imbalance.combine import SMOTETomek
smote=SMOTETomek(0.90)
y=df['Loan_Status']
x=df.drop(columns=['Loan_Status'],axis=1)
#creating a new x and y variables for the balanced set
x_bal,y_bal=smote.fit_resample(x,y)
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [56], in <cell line: 2>()
      1 #dividing the dataset into dependent and independent y and x respectively
----> 2 from imbalance.combine import SMOTETomek
      3 smote=SMOTETomek(0.90)
      4 y=df['Loan_Status']

ModuleNotFoundError: No module named 'imbalance'
```

In [58]:
```
#printing the values of y before balancing the data and after
print(y.value_counts())
print(y_bal.value_counts())
```

```
Y    422
N    192
Name: Loan_Status, dtype: int64
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [58], in <cell line: 3>()
      1 #printing the values of y before balancing the data and after
      2 print(y.value_counts())
----> 3 print(y_bal.value_counts())

NameError: name 'y_bal' is not defined
```

In [59]:
```
df.describe()
```

Out[59]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	614.000000	614.000000	614.000000
mean	5403.459283	1621.245798	145.465798	342.410423	0.855049
std	6109.041673	2926.248369	84.180967	64.428629	0.352339
min	150.000000	0.000000	9.000000	12.000000	0.000000
25%	2877.500000	0.000000	100.250000	360.000000	1.000000
50%	3812.500000	1188.500000	125.000000	360.000000	1.000000
75%	5795.000000	2297.250000	164.750000	360.000000	1.000000
max	81000.000000	41667.000000	700.000000	480.000000	1.000000

In [60]:
```
df1.describe()
```

Out[60]:

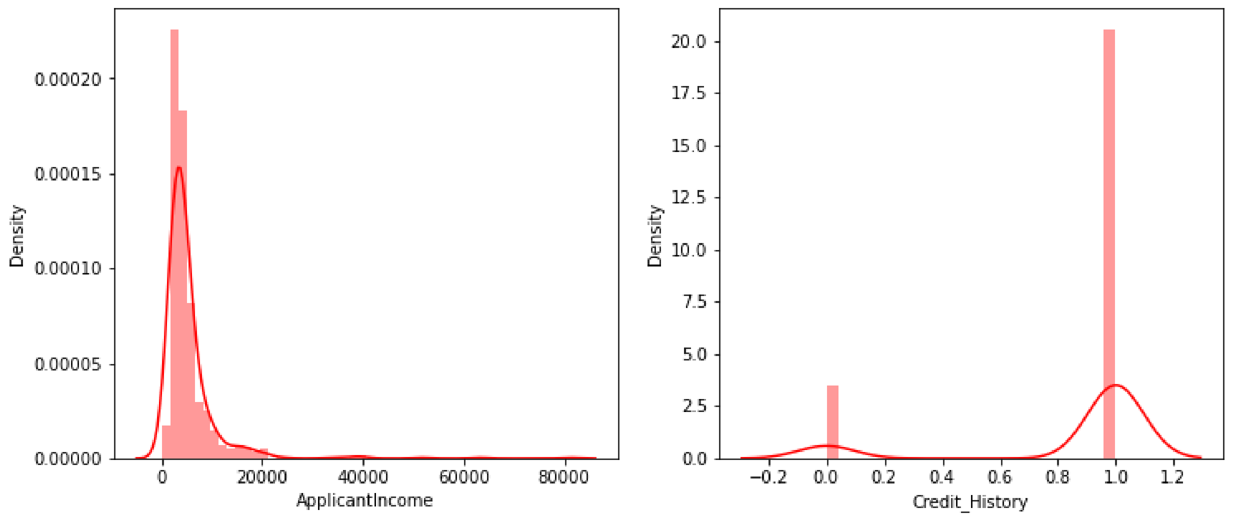
	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	367.000000	367.000000	367.000000	367.000000	367.000000
mean	4805.599455	1569.577657	136.321526	342.822888	0.839237
std	4910.685399	2334.232099	60.967295	64.658402	0.367814
min	0.000000	0.000000	28.000000	6.000000	0.000000
25%	2864.000000	0.000000	101.000000	360.000000	1.000000
50%	3786.000000	1025.000000	126.000000	360.000000	1.000000
75%	5060.000000	2430.500000	157.500000	360.000000	1.000000
max	72529.000000	24000.000000	550.000000	480.000000	1.000000

In [63]:
```
#Data Visualization using distplot
plt.figure(figsize = (12,5))
plt.subplot(121)
sns.distplot(df['ApplicantIncome'],color='r')
plt.subplot(122)
sns.distplot(df['Credit_History'],color='r')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[63]:

```
<AxesSubplot:xlabel='Credit_History', ylabel='Density'>
```



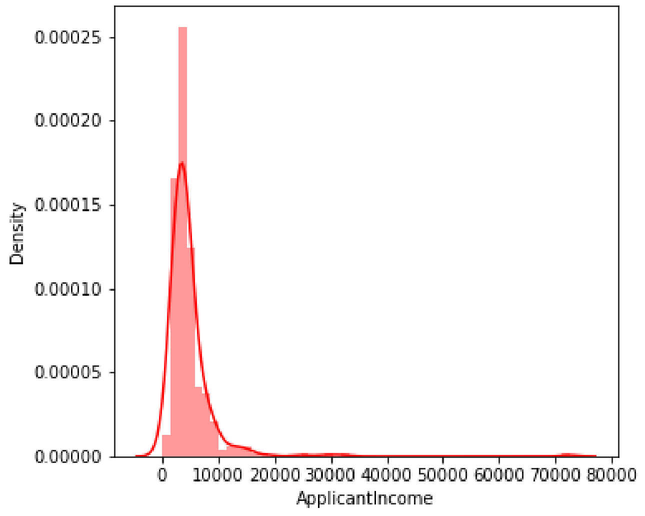
In [68]:

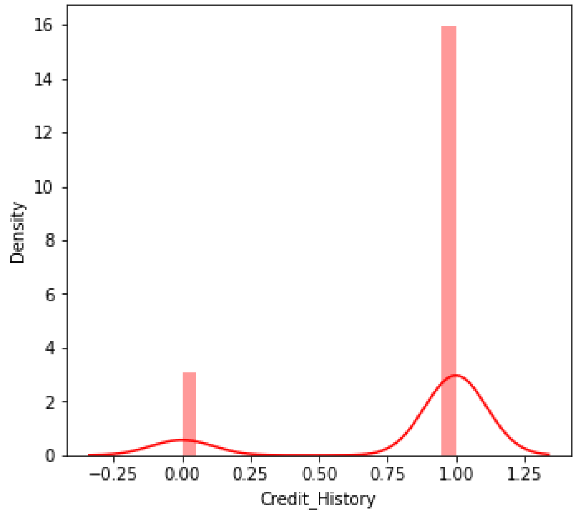
```python
#Data Visualization using distplot
plt.figure(figsize = (12,5))
plt.subplot(121)
sns.distplot(df1['ApplicantIncome'],color='r')
plt.subplot(122)
sns.distplot(df1['Credit_History'],color='r')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[68]:

```
<AxesSubplot:xlabel='Credit_History', ylabel='Density'>
```

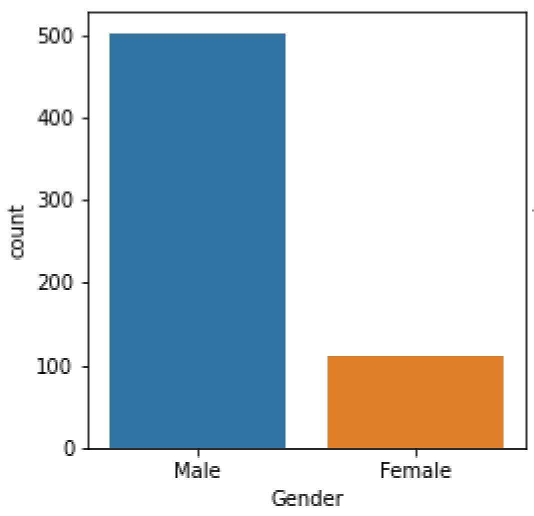


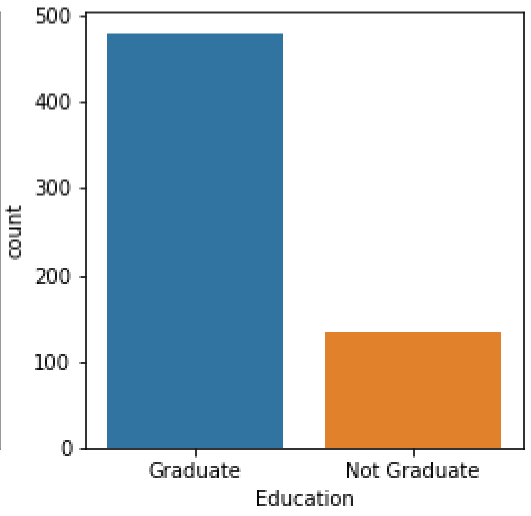


In [65]:
```
#Bivariate analysis
#Data Visualization using countplot
plt.figure(figsize = (18,4))
plt.subplot(1,4,1)
sns.countplot(df['Gender'])
plt.subplot(1,4,2)
sns.countplot(df['Education'])
plt.show
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

Out[65]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```





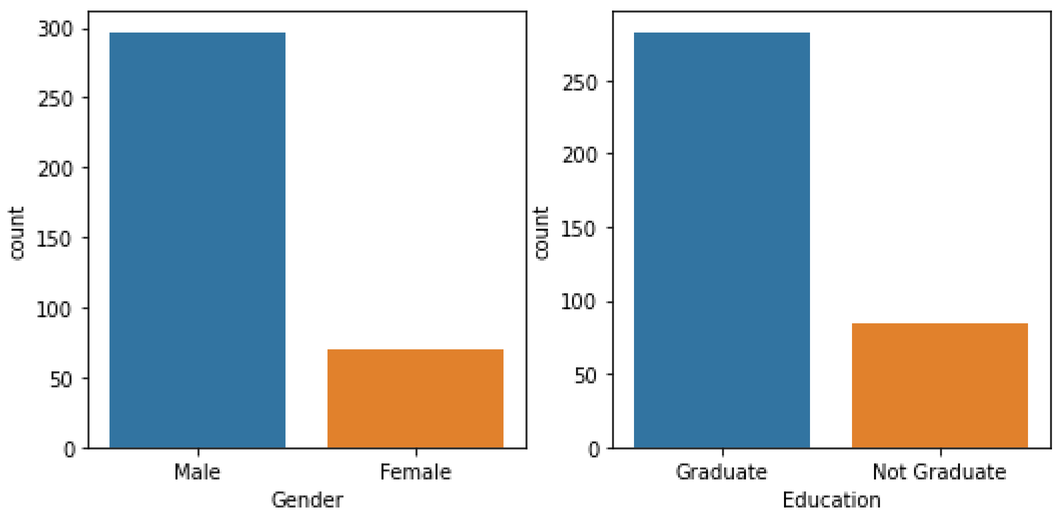
In [69]:
```
#Bivariate analysis
#Data Visualization using countplot
plt.figure(figsize = (18,4))
plt.subplot(1,4,1)
```

```
sns.countplot(df1['Gender'])
plt.subplot(1,4,2)
sns.countplot(df1['Education'])
plt.show
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

Out[69]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```

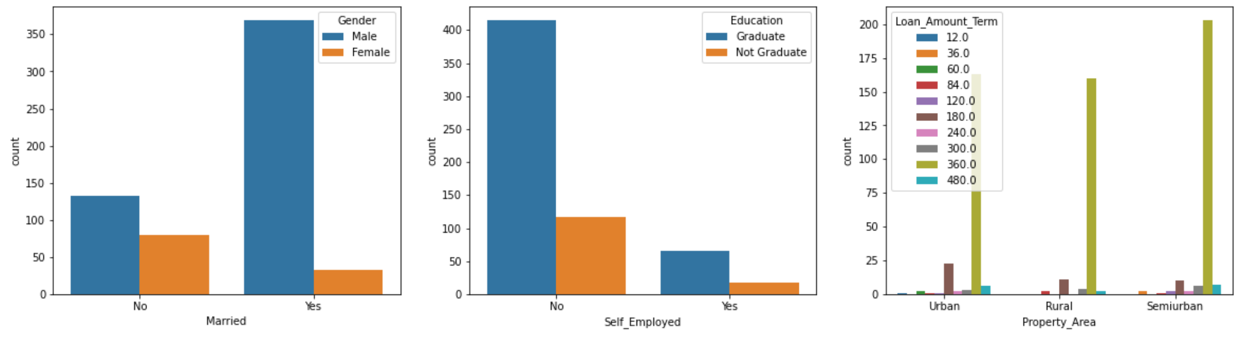


In [67]:

```
#Data Visualization using countplot
plt.figure(figsize = (20,5))
plt.subplot(131)
sns.countplot(df['Married'],hue=df['Gender'])
plt.subplot(132)
sns.countplot(df['Self_Employed'],hue=df['Education'])
plt.subplot(133)
sns.countplot(df['Property_Area'],hue=df['Loan_Amount_Term'])
plt.show
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

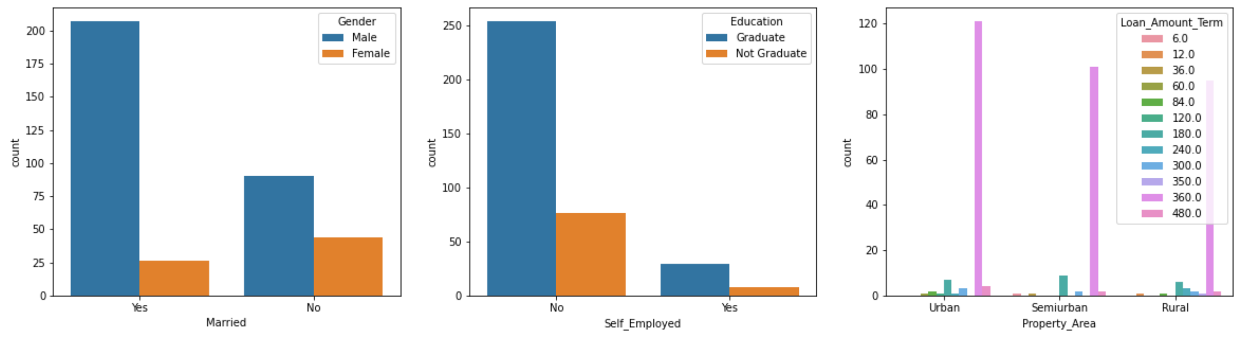
Out[67]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [70]:
```
#Data Visualization using countplot
plt.figure(figsize = (20,5))
plt.subplot(131)
sns.countplot(df1['Married'],hue=df1['Gender'])
plt.subplot(132)
sns.countplot(df1['Self_Employed'],hue=df1['Education'])
plt.subplot(133)
sns.countplot(df1['Property_Area'],hue=df1['Loan_Amount_Term'])
plt.show
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

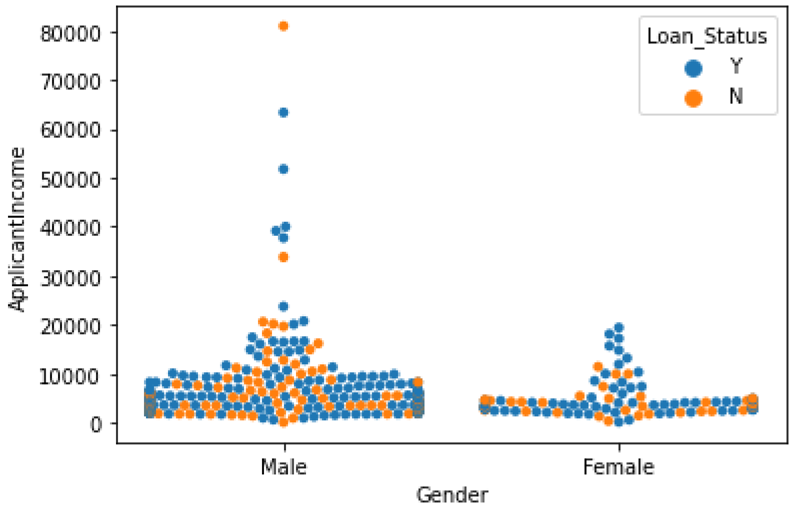
Out[70]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [72]:
```
#visualized based gender and income what would be the application status
sns.swarmplot(df['Gender'],df['ApplicantIncome'],hue=df['Loan_Status'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only valid
positional argument will be `data`, and passing other arguments without an explicit k
eyword will result in an error or misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning:
67.1% of the points cannot be placed; you may want to decrease the size of the marker
s or use stripplot.
  warnings.warn(msg, UserWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning:
33.0% of the points cannot be placed; you may want to decrease the size of the marker
s or use stripplot.
  warnings.warn(msg, UserWarning)
```

Out[72]:
```
<AxesSubplot:xlabel='Gender', ylabel='ApplicantIncome'>
```



In [73]:
```
#Sclaing the Data
#performing feature scaling operation using standard scaller an x part of the dataset
#because there different types of values in the columns
sc=StandardScalaer()
x_bal=sc.fit_transform(x_bal)
x_bal=pd.df(x_bal,columns=names)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [73], in <cell line: 4>()
      1 #Sclaing the Data
      2 #performing feature scaling operation using standard scaller an x part of the
dataset
      3 #because there different types of values in the columns
----> 4 sc=StandardScalaer()
      5 x_bal=sc.fit_transform(x_bal)
      6 x_bal=pd.df(x_bal,columns=names)

NameError: name 'StandardScalaer' is not defined
```

In [74]:
```
#Splitting the Dataset in train and test on balanced dataset
X_train,X_test,y_train,y_test=train_test_split(x_bal,y_bal,test_size=0.33,random_state
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [74], in <cell line: 2>()
      1 #Splitting the Dataset in train and test on balanced dataset
----> 2 X_train,X_test,y_train,y_test=train_test_split(x_bal,y_bal,test_size=0.33,ran
dom_state=42)

NameError: name 'x_bal' is not defined
```

In [75]:

```python
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [75], in <cell line: 5>()
      2 clf = DecisionTreeClassifier()
      4 # Train Decision Tree Classifer
----> 5 clf = clf.fit(X_train,y_train)
      7 #Predict the response for test dataset
      8 y_pred = clf.predict(X_test)

NameError: name 'X_train' is not defined
```

In [76]:

```python
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred)
```

```
  Input In [76]
    print("Accuracy:",metrics.accuracy_score(y_test, y_pred)
                                                             ^
SyntaxError: unexpected EOF while parsing
```

In [78]:

```
pip install graphviz

pip install pydotplus
```

```
  Input In [78]
    pip install graphviz
        ^
SyntaxError: invalid syntax
```

In [79]:

```python
from sklearn.tree import export_graphviz
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus

dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = feature_cols,class_names=['0',
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [79], in <cell line: 2>()
      1 from sklearn.tree import export_graphviz
----> 2 from sklearn.externals.six import StringIO
      3 from IPython.display import Image
      4 import pydotplus

ModuleNotFoundError: No module named 'sklearn.externals.six'
```

In [80]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=1000, n_features=4,n_informative=2, n_redundant=
clf = RandomForestClassifier(max_depth=2, random_state=0)
clf.fit(X, y)
RandomForestClassifier()
print(clf.predict([[0, 0, 0, 0]]))
```

```
[1]
```

In [81]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import sklearn
```

In [83]:
```
dff=pd.read_csv("E:\\NMDS\pl_train.csv")
X = dff.iloc[:, [1, 2, 3]].values
y = dff.iloc[:, -1].values
```

In [86]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_sta
```

In [87]:
```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [87], in <cell line: 4>()
      2 from sklearn.preprocessing import StandardScaler
      3 sc = StandardScaler()
----> 4 X_train = sc.fit_transform(X_train)
      5 X_test = sc.transform(X_test)

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:852, in TransformerMi
xin.fit_transform(self, X, y, **fit_params)
    848 # non-optimized default implementation; override when a better
    849 # method is possible for a given clustering algorithm
    850 if y is None:
    851     # fit method of arity 1 (unsupervised transformation)
--> 852     return self.fit(X, **fit_params).transform(X)
    853 else:
    854     # fit method of arity 2 (supervised transformation)
    855     return self.fit(X, y, **fit_params).transform(X)

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:806, i
n StandardScaler.fit(self, X, y, sample_weight)
    804 # Reset internal state before fitting
    805 self._reset()
--> 806 return self.partial_fit(X, y, sample_weight)

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:841, i
n StandardScaler.partial_fit(self, X, y, sample_weight)
    809 """Online computation of mean and std on X for later scaling.
    810 
    811 All of X is processed as a single batch. This is intended for cases
   (...)
    838     Fitted scaler.
    839 """
    840 first_call = not hasattr(self, "n_samples_seen_")
--> 841 X = self._validate_data(
    842     X,
    843     accept_sparse=("csr", "csc"),
    844     estimator=self,
    845     dtype=FLOAT_DTYPES,
    846     force_all_finite="allow-nan",
    847     reset=first_call,
    848 )
    849 n_features = X.shape[1]
    851 if sample_weight is not None:

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:566, in BaseEstimato
r._validate_data(self, X, y, reset, validate_separately, **check_params)
    564     raise ValueError("Validation should be done on X, y or both.")
    565 elif not no_val_X and no_val_y:
--> 566     X = check_array(X, **check_params)
    567     out = X
    568 elif no_val_X and not no_val_y:

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:746, in c
heck_array(array, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_f
inite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    744         array = array.astype(dtype, casting="unsafe", copy=False)
    745     else:
--> 746         array = np.asarray(array, order=order, dtype=dtype)
    747 except ComplexWarning as complex_warning:
```

```
    748     raise ValueError(
    749         "Complex data not supported\n{}\n".format(array)
    750     ) from complex_warning

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

In [88]:
```
# Training the K-NN model on the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [88], in <cell line: 4>()
      2 from sklearn.neighbors import KNeighborsClassifier
      3 classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p =
2)
----> 4 classifier.fit(X_train, y_train)

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
198, in KNeighborsClassifier.fit(self, X, y)
    179 """Fit the k-nearest neighbors classifier from the training dataset.
    180 
    181 Parameters
   (...)
    194     The fitted k-nearest neighbors classifier.
    195 """
    196 self.weights = _check_weights(self.weights)
--> 198 return self._fit(X, y)

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_base.py:400, in Ne
ighborsBase._fit(self, X, y)
    398 if self._get_tags()["requires_y"]:
    399     if not isinstance(X, (KDTree, BallTree, NeighborsBase)):
--> 400         X, y = self._validate_data(X, y, accept_sparse="csr", multi_output=Tr
ue)
    402     if is_classifier(self):
    403         # Classification targets require a specific format
    404         if y.ndim == 1 or y.ndim == 2 and y.shape[1] == 1:

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:581, in BaseEstimato
r._validate_data(self, X, y, reset, validate_separately, **check_params)
    579         y = check_array(y, **check_y_params)
    580     else:
--> 581         X, y = check_X_y(X, y, **check_params)
    582     out = X, y
    584 if not no_val_X and check_params.get("ensure_2d", True):

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:964, in c
heck_X_y(X, y, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_fini
te, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_features, y_num
eric, estimator)
    961 if y is None:
    962     raise ValueError("y cannot be None")
--> 964 X = check_array(
    965     X,
    966     accept_sparse=accept_sparse,
    967     accept_large_sparse=accept_large_sparse,
    968     dtype=dtype,
    969     order=order,
    970     copy=copy,
    971     force_all_finite=force_all_finite,
    972     ensure_2d=ensure_2d,
    973     allow_nd=allow_nd,
    974     ensure_min_samples=ensure_min_samples,
    975     ensure_min_features=ensure_min_features,
    976     estimator=estimator,
    977 )
    979 y = _check_y(y, multi_output=multi_output, y_numeric=y_numeric)
    981 check_consistent_length(X, y)
```

```
File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:746, in c
heck_array(array, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_f
inite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    744         array = array.astype(dtype, casting="unsafe", copy=False)
    745     else:
--> 746         array = np.asarray(array, order=order, dtype=dtype)
    747 except ComplexWarning as complex_warning:
    748     raise ValueError(
    749         "Complex data not supported\n{}\n".format(array)
    750     ) from complex_warning

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

In [89]:

```
# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
```

```
---------------------------------------------------------------------------
NotFittedError                            Traceback (most recent call last)
Input In [89], in <cell line: 2>()
      1 # Predicting the Test set results
----> 2 y_pred = classifier.predict(X_test)
      4 # Making the Confusion Matrix
      5 from sklearn.metrics import confusion_matrix, accuracy_score

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
214, in KNeighborsClassifier.predict(self, X)
    200 def predict(self, X):
    201     """Predict the class labels for the provided data.
    202 
    203     Parameters
   (...)
    212         Class labels for each data sample.
    213     """
--> 214     neigh_dist, neigh_ind = self.kneighbors(X)
    215     classes_ = self.classes_
    216     _y = self._y

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_base.py:700, in KN
eighborsMixin.kneighbors(self, X, n_neighbors, return_distance)
    647 def kneighbors(self, X=None, n_neighbors=None, return_distance=True):
    648     """Find the K-neighbors of a point.
    649 
    650     Returns indices of and distances to the neighbors of each point.
   (...)
    698             [2]]...)
    699     """
--> 700     check_is_fitted(self)
    702     if n_neighbors is None:
    703         n_neighbors = self.n_neighbors

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:1222, in
check_is_fitted(estimator, attributes, msg, all_or_any)
   1217     fitted = [
   1218         v for v in vars(estimator) if v.endswith("_") and not v.startswith("_
_")
   1219     ]
   1221 if not fitted:
-> 1222     raise NotFittedError(msg % {"name": type(estimator).__name__})

NotFittedError: This KNeighborsClassifier instance is not fitted yet. Call 'fit' with
appropriate arguments before using this estimator.
```

In [ ]: