loan-approvalpred

October 17, 2023

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
df = pd.read_csv(r"C:\Users\amits\Downloads\loan_approval_dataset.csv")
```

[3]:
```
df.head()
```

[3]:

	loan_id	no_of_dependents	education	self_employed	income_annum
0	1	2	Graduate	No	9600000
1	2	0	Not Graduate	Yes	4100000
2	3	3	Graduate	No	9100000
3	4	3	Graduate	No	8200000
4	5	5	Not Graduate	Yes	9800000

	loan_amount	loan_term	cibil_score	residential_assets_value
0	29900000	12	778	2400000
1	12200000	8	417	2700000
2	29700000	20	506	7100000
3	30700000	8	467	18200000
4	24200000	20	382	12400000

	commercial_assets_value	luxury_assets_value	bank_asset_value
0	17600000	22700000	8000000
1	2200000	8800000	3300000
2	4500000	33300000	12800000
3	3300000	23300000	7900000
4	8200000	29400000	5000000

	loan_status
0	Approved
1	Rejected
2	Rejected
3	Rejected
4	Rejected

[4]:
```
df.shape
```

[4]: (4269, 13)

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4269 entries, 0 to 4268
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   loan_id                   4269 non-null   int64
 1    no_of_dependents         4269 non-null   int64
 2    education                4269 non-null   object
 3    self_employed            4269 non-null   object
 4    income_annum             4269 non-null   int64
 5    loan_amount              4269 non-null   int64
 6    loan_term                4269 non-null   int64
 7    cibil_score              4269 non-null   int64
 8    residential_assets_value 4269 non-null   int64
 9    commercial_assets_value  4269 non-null   int64
 10   luxury_assets_value      4269 non-null   int64
 11   bank_asset_value         4269 non-null   int64
 12   loan_status              4269 non-null   object
dtypes: int64(10), object(3)
memory usage: 433.7+ KB
```

[6]:
```
df.describe()
```

[6]:

	loan_id	no_of_dependents	income_annum	loan_amount	\
count	4269.000000	4269.000000	4.269000e+03	4.269000e+03	
mean	2135.000000	2.498712	5.059124e+06	1.513345e+07	
std	1232.498479	1.695910	2.806840e+06	9.043363e+06	
min	1.000000	0.000000	2.000000e+05	3.000000e+05	
25%	1068.000000	1.000000	2.700000e+06	7.700000e+06	
50%	2135.000000	3.000000	5.100000e+06	1.450000e+07	
75%	3202.000000	4.000000	7.500000e+06	2.150000e+07	
max	4269.000000	5.000000	9.900000e+06	3.950000e+07	

	loan_term	cibil_score	residential_assets_value	\
count	4269.000000	4269.000000	4.269000e+03	
mean	10.900445	599.936051	7.472617e+06	
std	5.709187	172.430401	6.503637e+06	
min	2.000000	300.000000	-1.000000e+05	
25%	6.000000	453.000000	2.200000e+06	
50%	10.000000	600.000000	5.600000e+06	
75%	16.000000	748.000000	1.130000e+07	

```
max      20.000000    900.000000               2.910000e+07

       commercial_assets_value   luxury_assets_value   bank_asset_value
count             4.269000e+03          4.269000e+03       4.269000e+03
mean              4.973155e+06          1.512631e+07       4.976692e+06
std               4.388966e+06          9.103754e+06       3.250185e+06
min               0.000000e+00          3.000000e+05       0.000000e+00
25%               1.300000e+06          7.500000e+06       2.300000e+06
50%               3.700000e+06          1.460000e+07       4.600000e+06
75%               7.600000e+06          2.170000e+07       7.100000e+06
max               1.940000e+07          3.920000e+07       1.470000e+07
```

[7]:
```
# Exploratory Data Analysis
```

[8]:
```
df.columns
```

[8]:
```
Index(['loan_id', ' no_of_dependents', ' education', ' self_employed',
       ' income_annum', ' loan_amount', ' loan_term', ' cibil_score',
       ' residential_assets_value', ' commercial_assets_value',
       ' luxury_assets_value', ' bank_asset_value', ' loan_status'],
      dtype='object')
```

[9]:
```
col_names = [col.strip() for col in df.columns]
df.columns = col_names
```

[10]:
```
df.columns
```

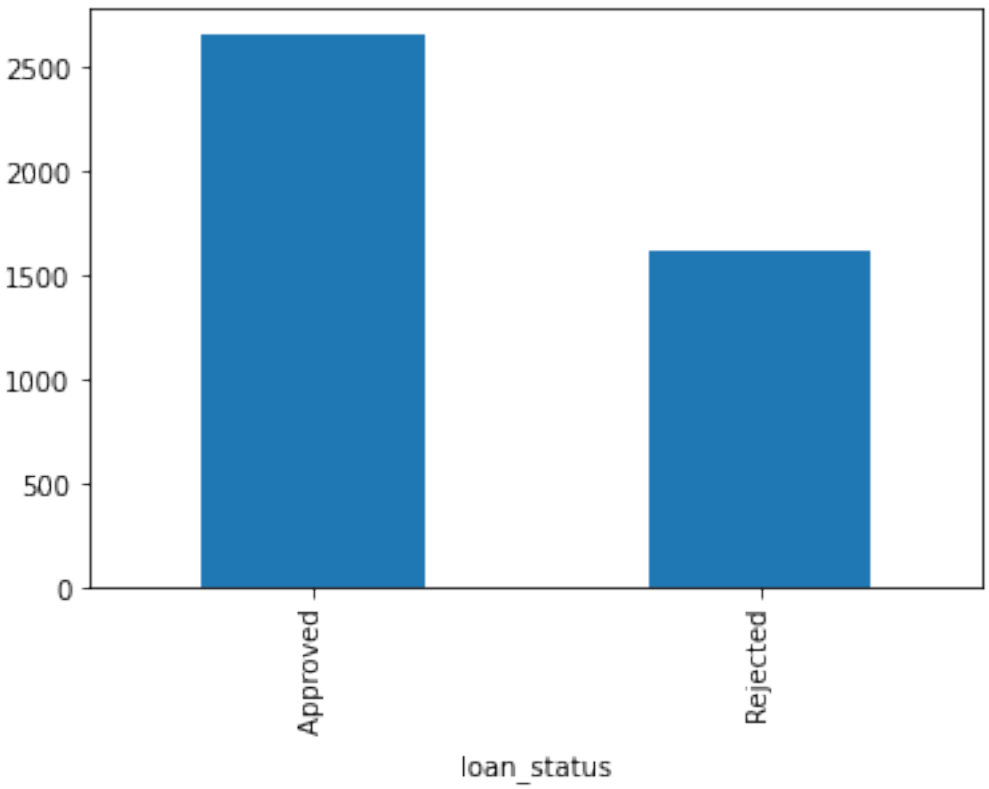
[10]:
```
Index(['loan_id', 'no_of_dependents', 'education', 'self_employed',
       'income_annum', 'loan_amount', 'loan_term', 'cibil_score',
       'residential_assets_value', 'commercial_assets_value',
       'luxury_assets_value', 'bank_asset_value', 'loan_status'],
      dtype='object')
```

[11]:
```
df['loan_status'].value_counts()
```

[11]:
```
loan_status
 Approved    2656
 Rejected    1613
Name: count, dtype: int64
```

[12]:
```
df['loan_status'].value_counts().plot(kind = "bar")
```

[12]:
```
<Axes: xlabel='loan_status'>
```

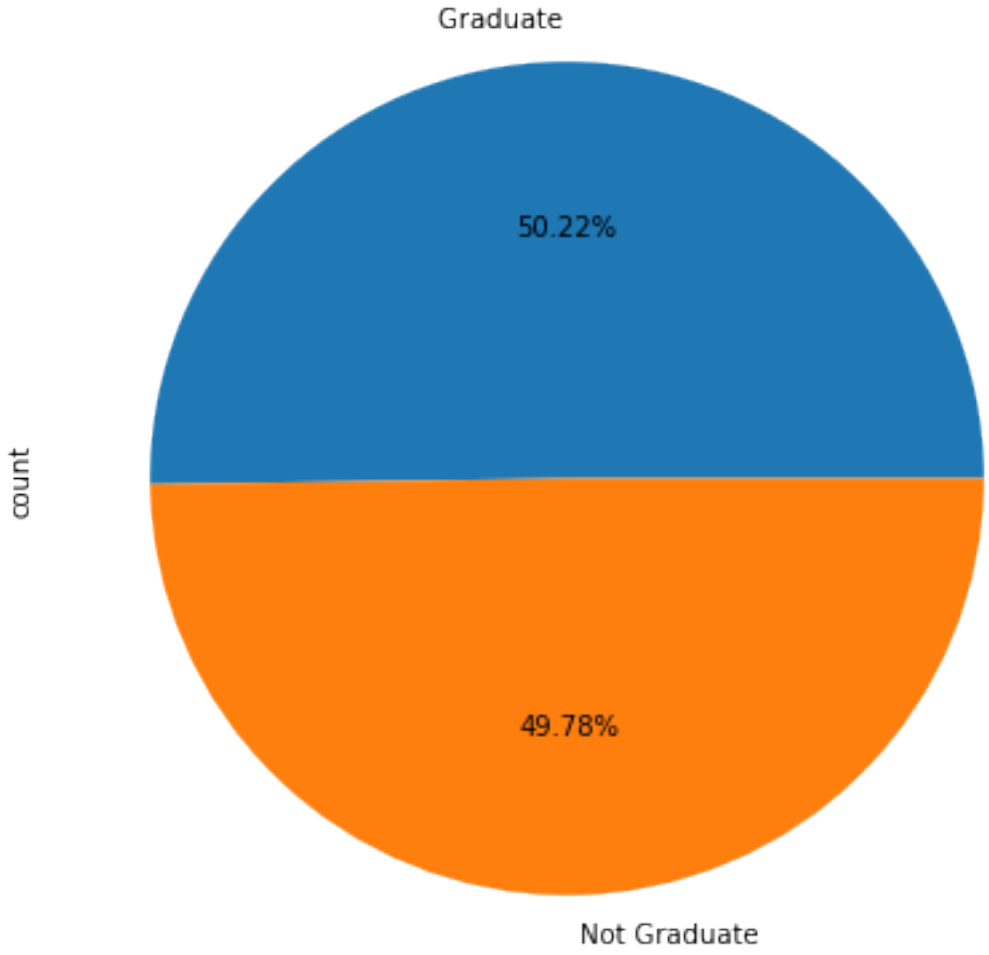


[13]:
```
df["education"].value_counts()
```

[13]:
```
education
 Graduate        2144
 Not Graduate    2125
Name: count, dtype: int64
```

[14]:
```
df["education"].value_counts().plot(kind = 'pie', autopct = "%1.2f%%", figsize=␣
↪(7,8))
```

[14]:
```
<Axes: ylabel='count'>
```

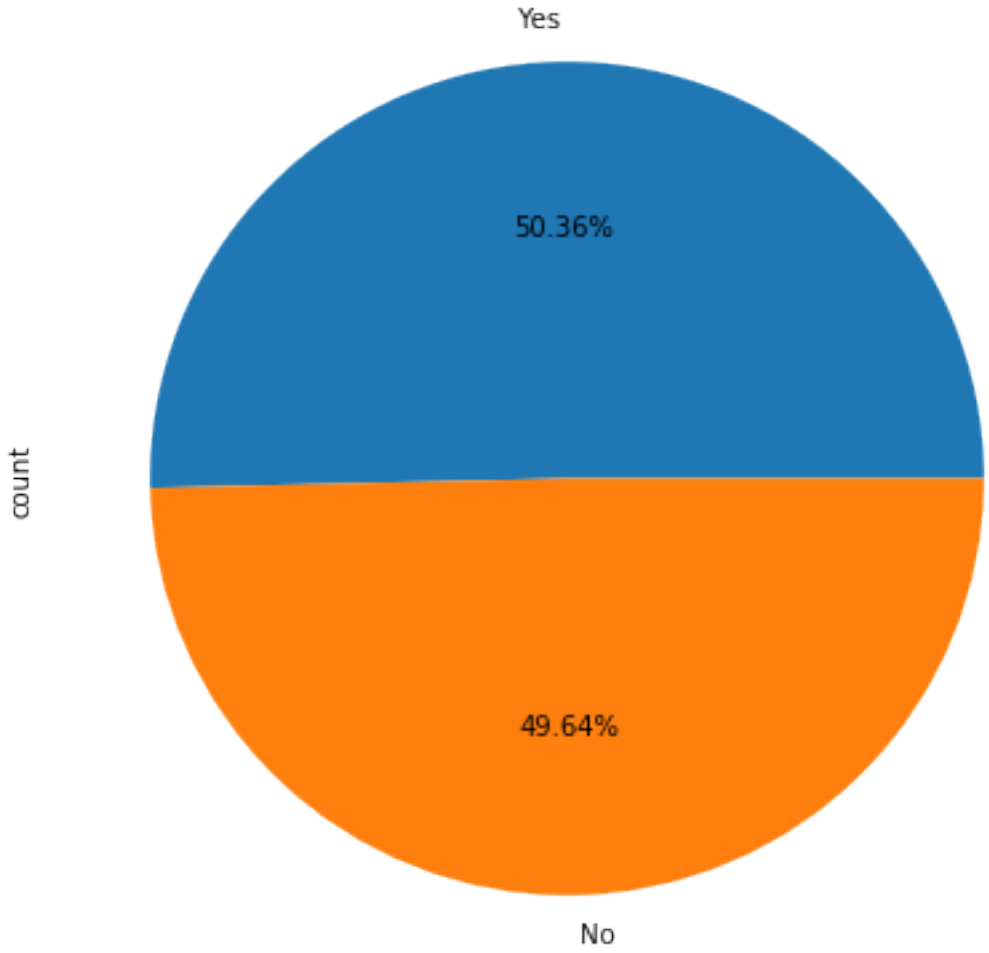


[15]:
```
df["self_employed"].value_counts()
```

[15]:
```
self_employed
 Yes    2150
 No     2119
Name: count, dtype: int64
```

[16]:
```
df["self_employed"].value_counts().plot(kind = 'pie', autopct = "%1.2f%%",␣
 ↪figsize= (7,8))
```

[16]:
```
<Axes: ylabel='count'>
```



[17]:
```
df.head()
```

[17]:

	loan_id	no_of_dependents	education	self_employed	income_annum	loan_amount	loan_term	cibil_score	residential_assets_value
0	1	2	Graduate	No	9600000	29900000	12	778	2400000
1	2	0	Not Graduate	Yes	4100000	12200000	8	417	2700000
2	3	3	Graduate	No	9100000	29700000	20	506	7100000
3	4	3	Graduate	No	8200000	30700000	8	467	18200000
4	5	5	Not Graduate	Yes	9800000				

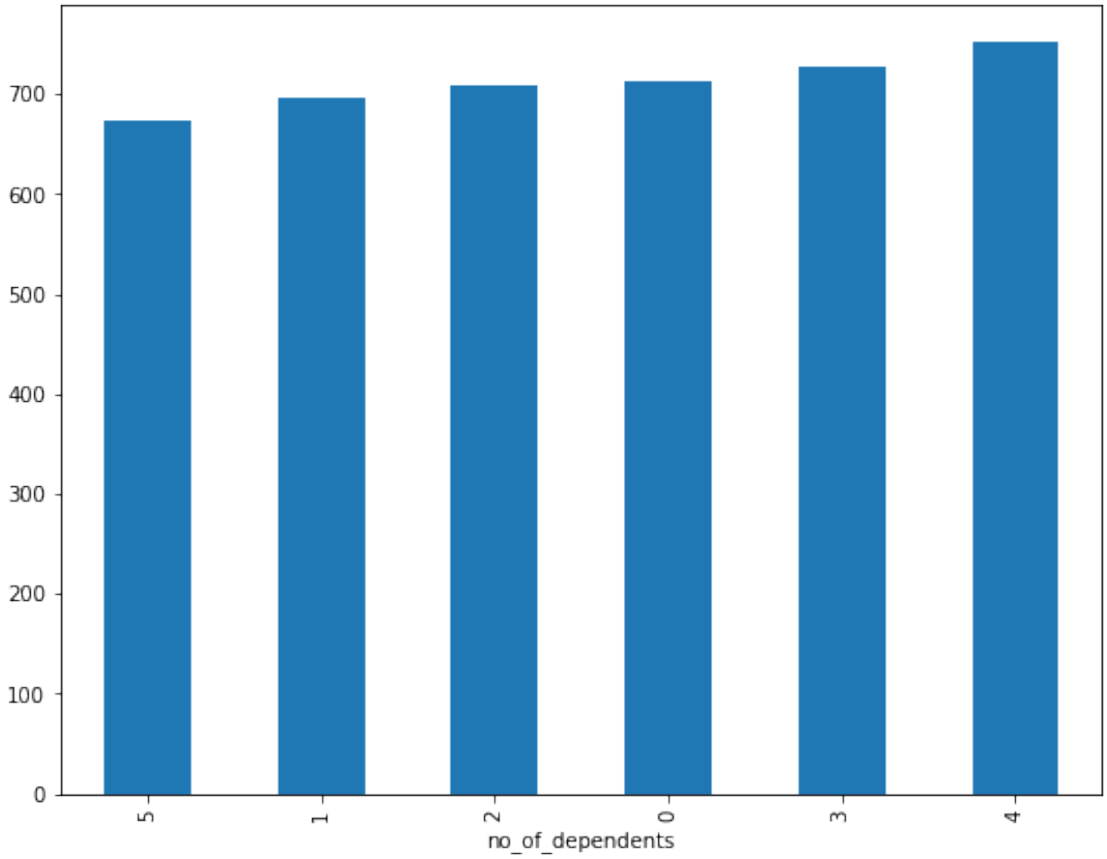
```
4     24200000          20           382                  12400000

   commercial_assets_value  luxury_assets_value  bank_asset_value loan_status
0                 17600000             22700000           8000000    Approved
1                  2200000              8800000           3300000    Rejected
2                  4500000             33300000          12800000    Rejected
3                  3300000             23300000           7900000    Rejected
4                  8200000             29400000           5000000    Rejected
```

[18]:
```
df['no_of_dependents'].value_counts()
```

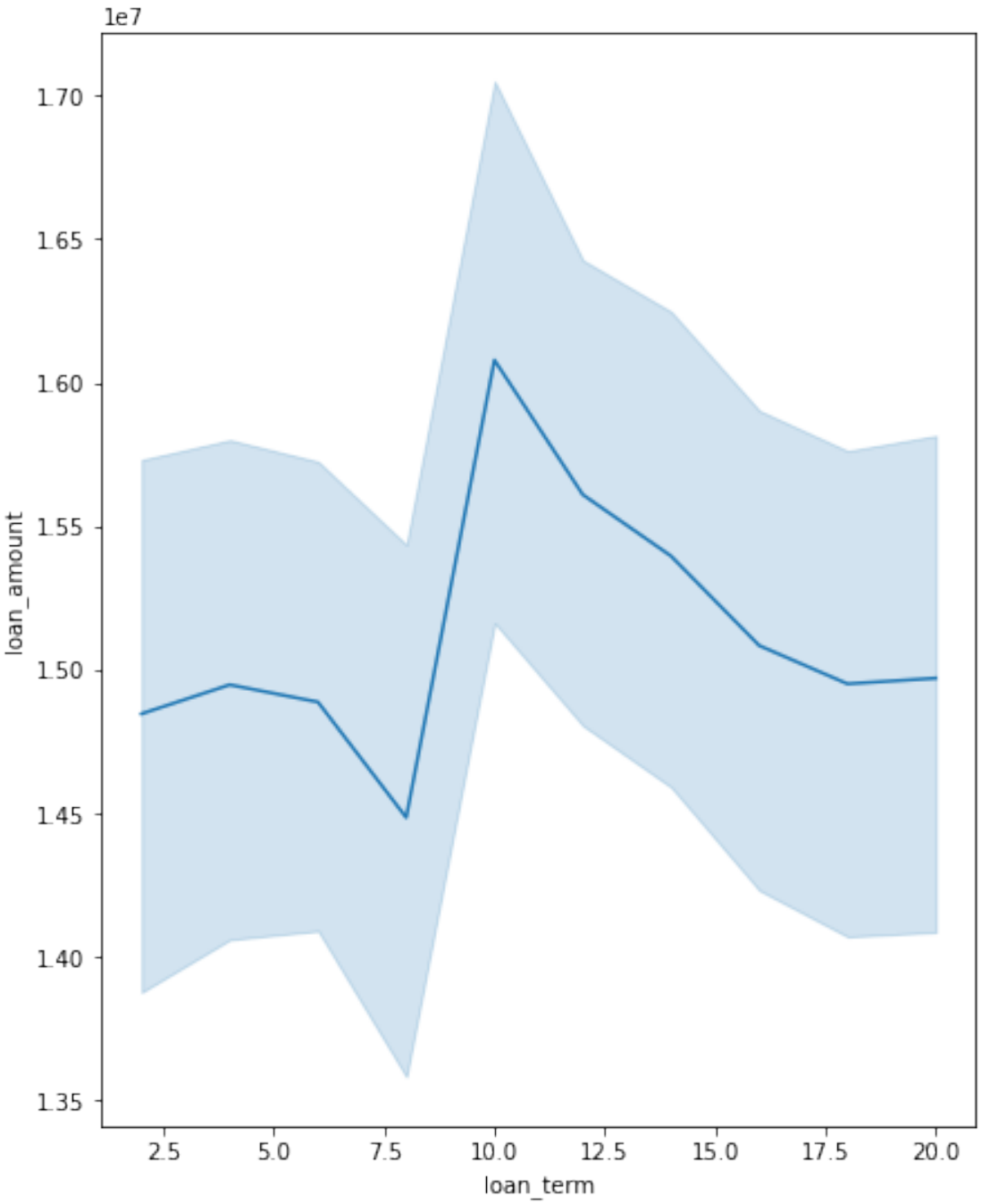
[18]:
```
no_of_dependents
4    752
3    727
0    712
2    708
1    697
5    673
Name: count, dtype: int64
```

[19]:
```
df['no_of_dependents'].value_counts().sort_values(ascending = True).plot(kind =␣
 ↪'bar', figsize = (9,7))
```

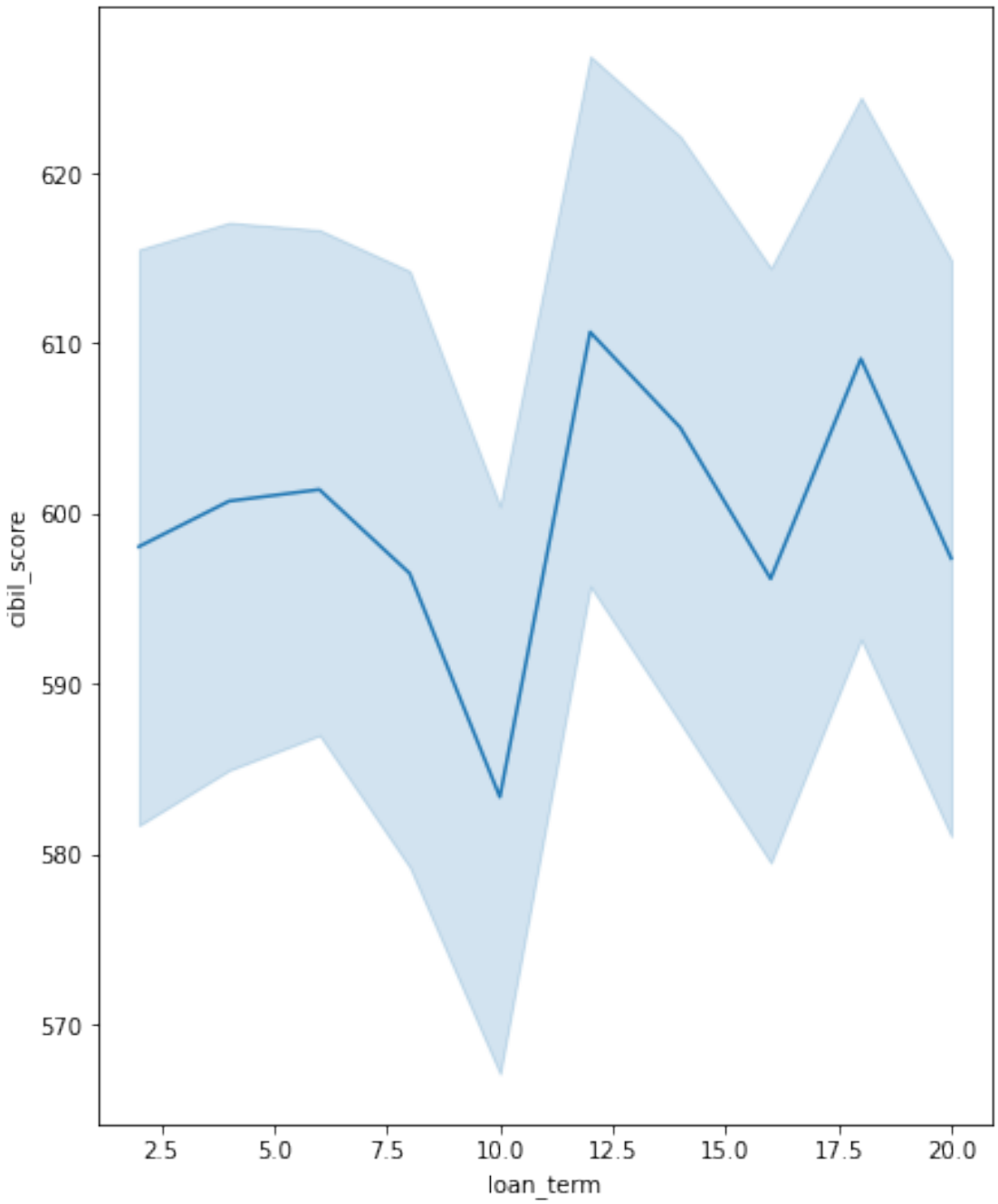
[19]:
```
<Axes: xlabel='no_of_dependents'>
```



[20]:
```
plt.figure(figsize = (7,9))
sns.lineplot(x=('loan_term'), y = ('loan_amount'), data=df)
plt.show()
```



[21]:
```
plt.figure(figsize = (7,9))
sns.lineplot(x=('loan_term'), y = ('cibil_score'), data=df)
plt.show()
```



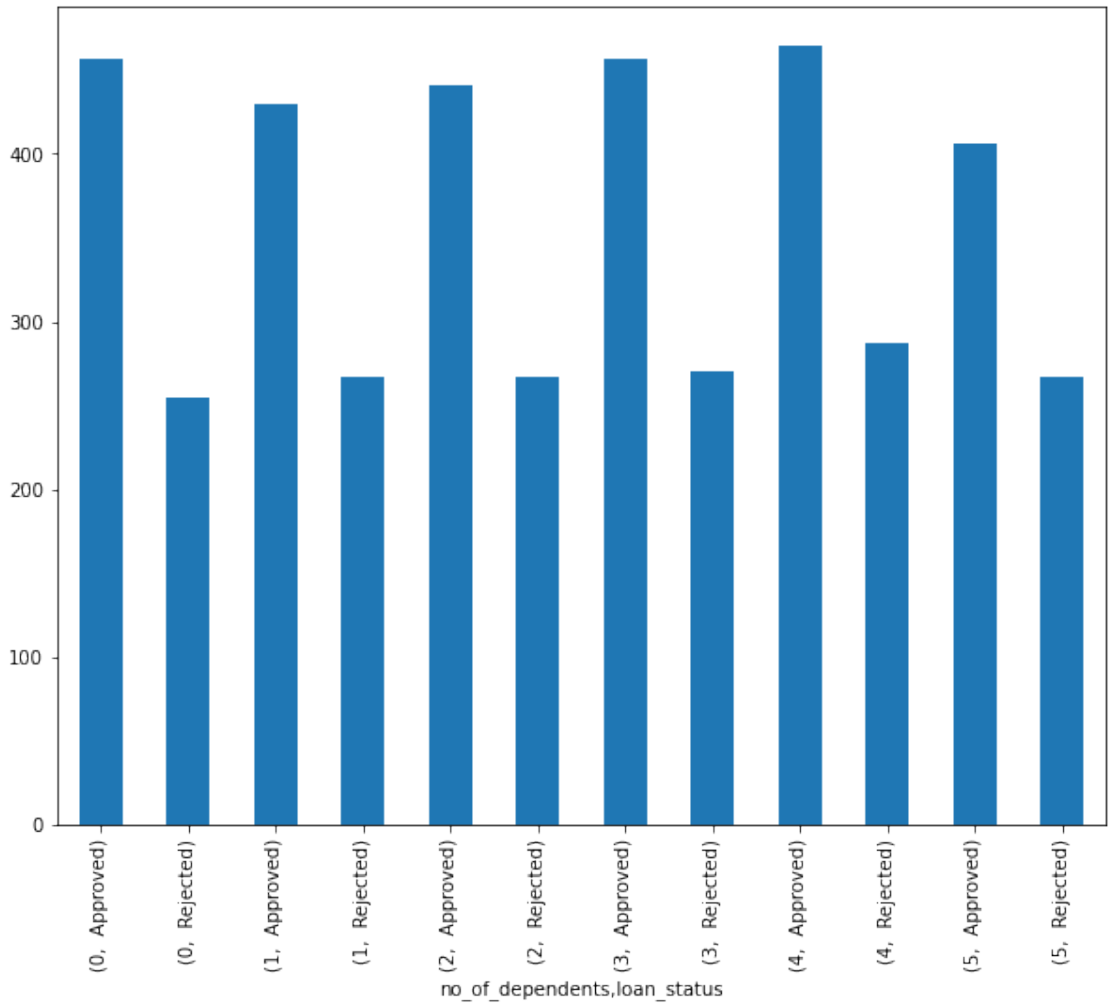
[22]:
```
df.head()
```

[22]:

	loan_id	no_of_dependents	education	self_employed	income_annum	\
0	1	2	Graduate	No	9600000	
1	2	0	Not Graduate	Yes	4100000	
2	3	3	Graduate	No	9100000	
3	4	3	Graduate	No	8200000	

```
4         5                5   Not Graduate           Yes        9800000

   loan_amount  loan_term  cibil_score  residential_assets_value  \
0     29900000         12          778                   2400000
1     12200000          8          417                   2700000
2     29700000         20          506                   7100000
3     30700000          8          467                  18200000
4     24200000         20          382                  12400000

   commercial_assets_value  luxury_assets_value  bank_asset_value loan_status
0                 17600000             22700000           8000000    Approved
1                  2200000              8800000           3300000    Rejected
2                  4500000             33300000          12800000    Rejected
3                  3300000             23300000           7900000    Rejected
4                  8200000             29400000           5000000    Rejected
```

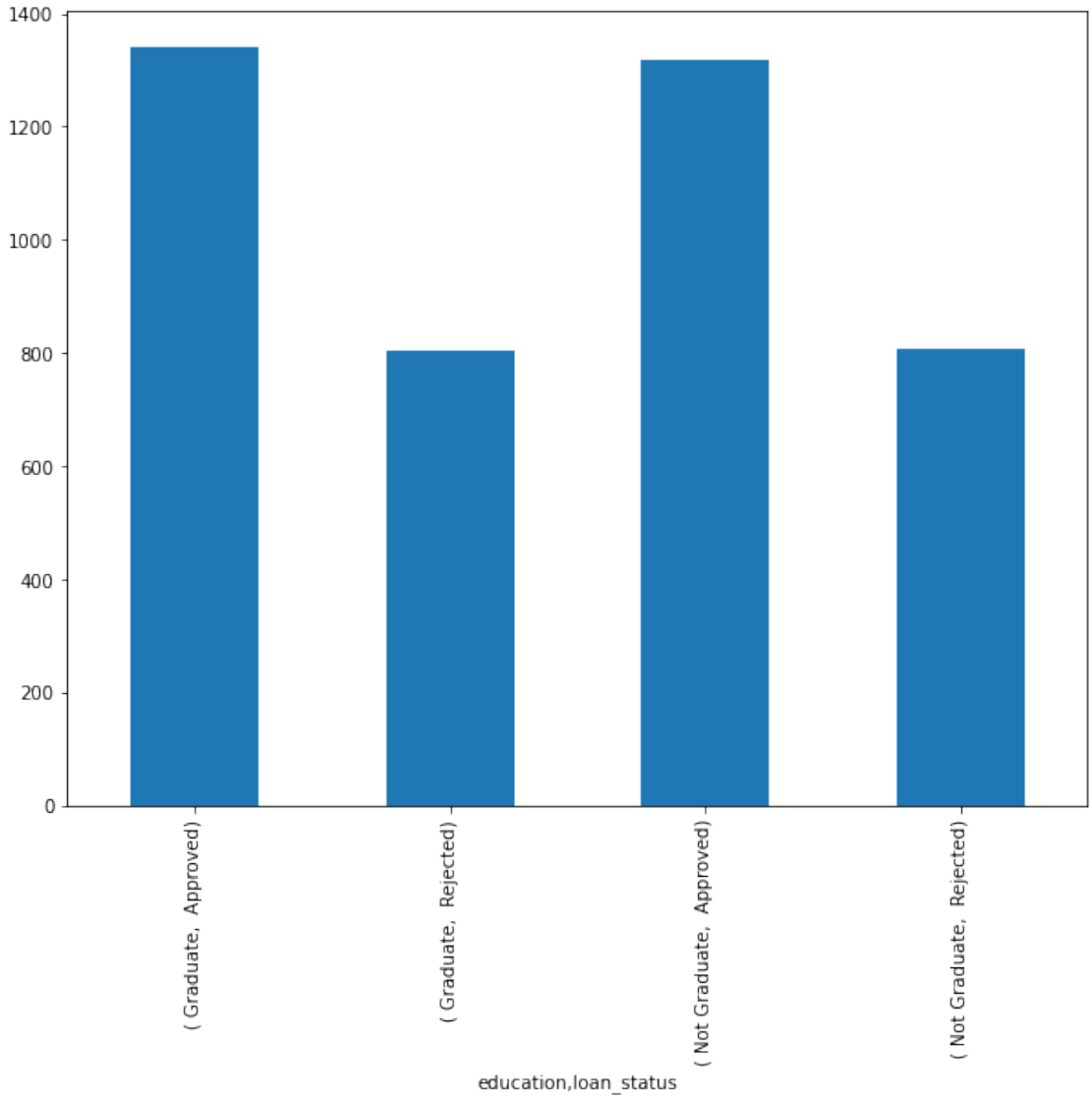
[23]:
```
df.groupby('no_of_dependents')['loan_status'].value_counts().plot(kind = 'bar',␣
↪figsize=(10,8))
plt.show()
```



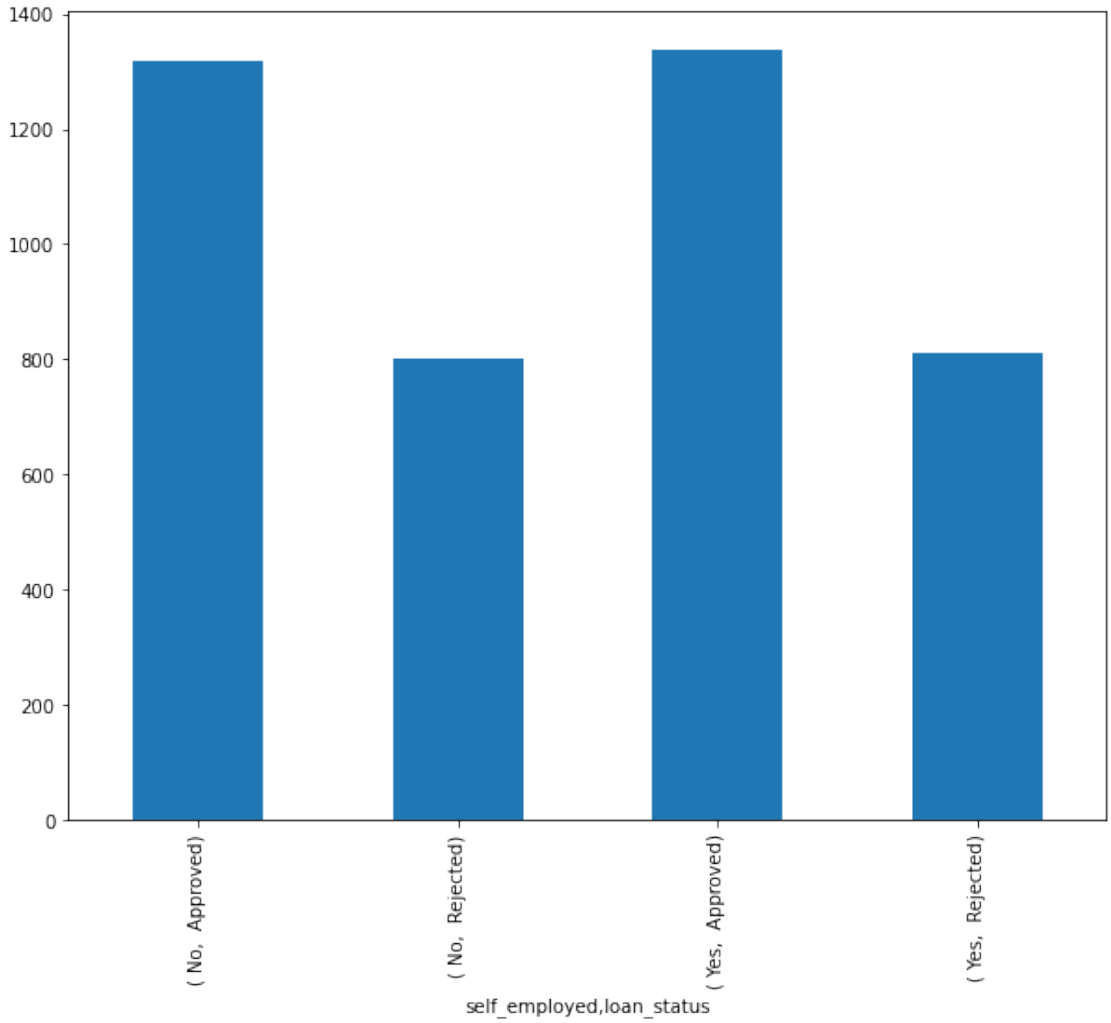
[24]:
```
df.groupby('education')['loan_status'].value_counts()
```

[24]:
```
education      loan_status
 Graduate       Approved      1339
                Rejected       805
 Not Graduate   Approved      1317
                Rejected       808
Name: count, dtype: int64
```

[25]:
```
df.groupby('education')['loan_status'].value_counts().plot(kind = 'bar',␣
  ↪figsize=(10,8))
plt.show()
```

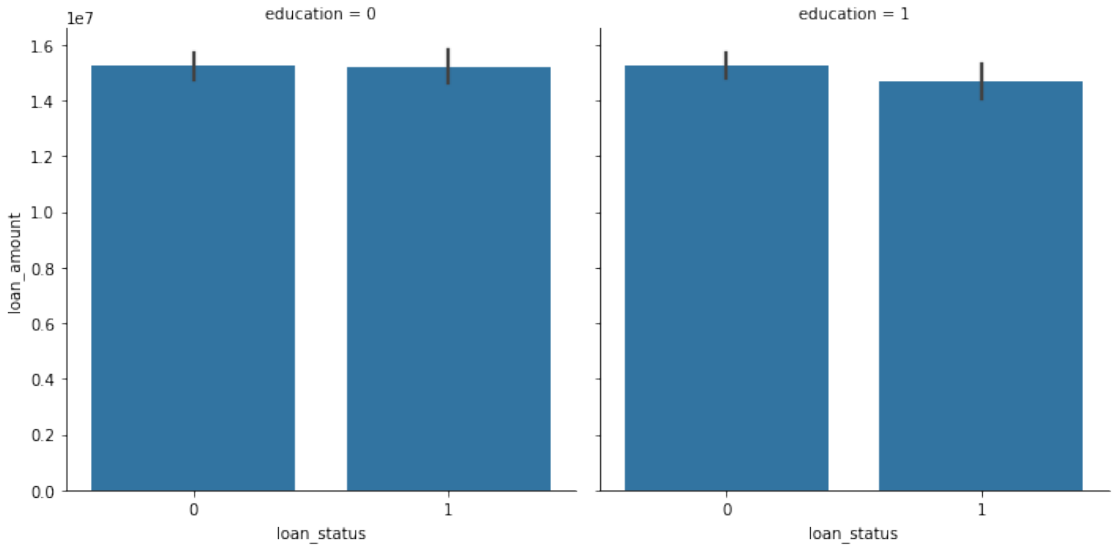


[26]:
```
df.groupby('self_employed')['loan_status'].value_counts().plot(kind = 'bar',␣
↪figsize=(10,8))
plt.show()
```



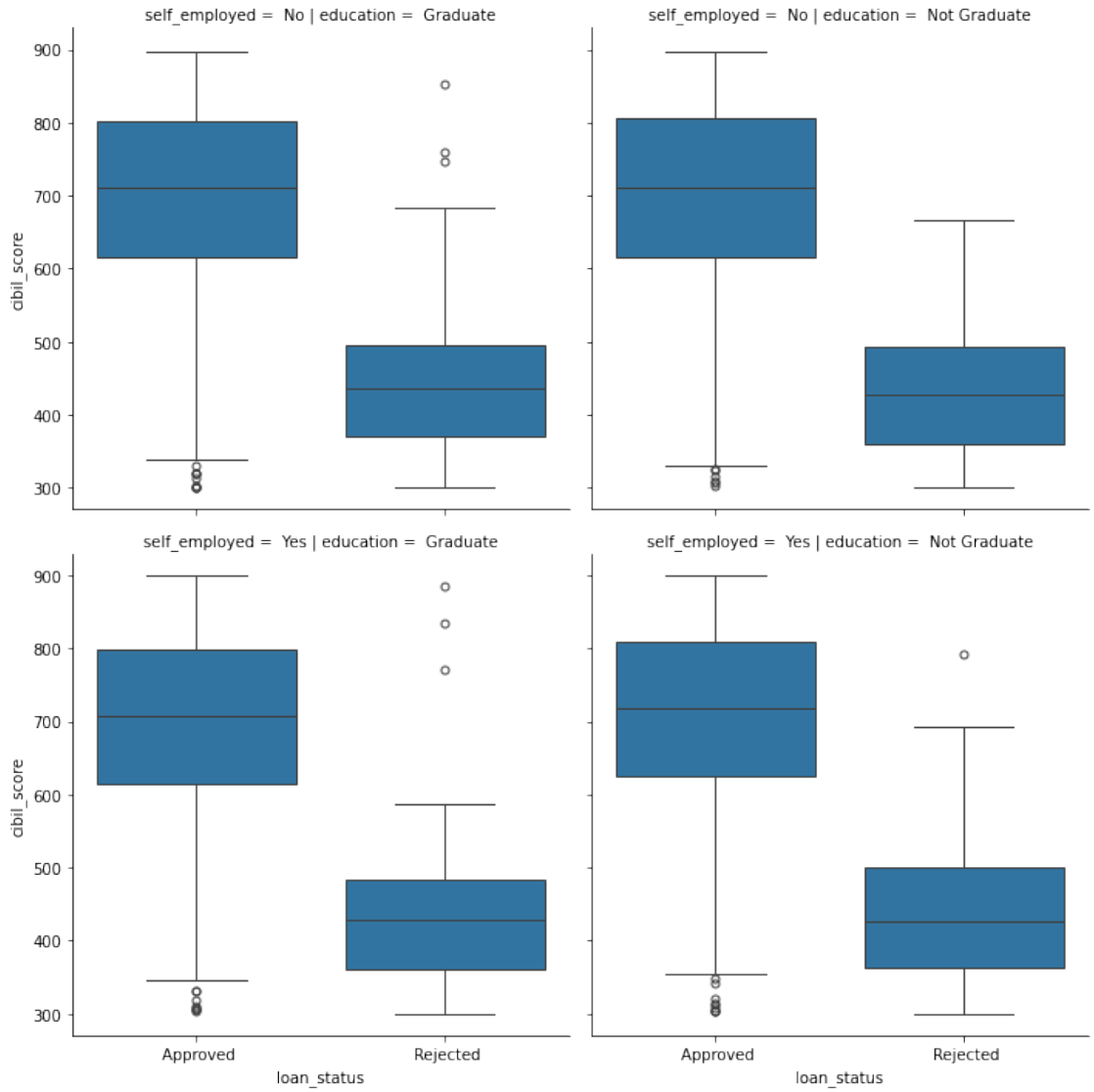
[50]:
```
sns.catplot(x = df['loan_status'], y= df['loan_amount'], kind = 'bar', col =␣
↪df['education'])
```

[50]: <seaborn.axisgrid.FacetGrid at 0x1ff3628f160>



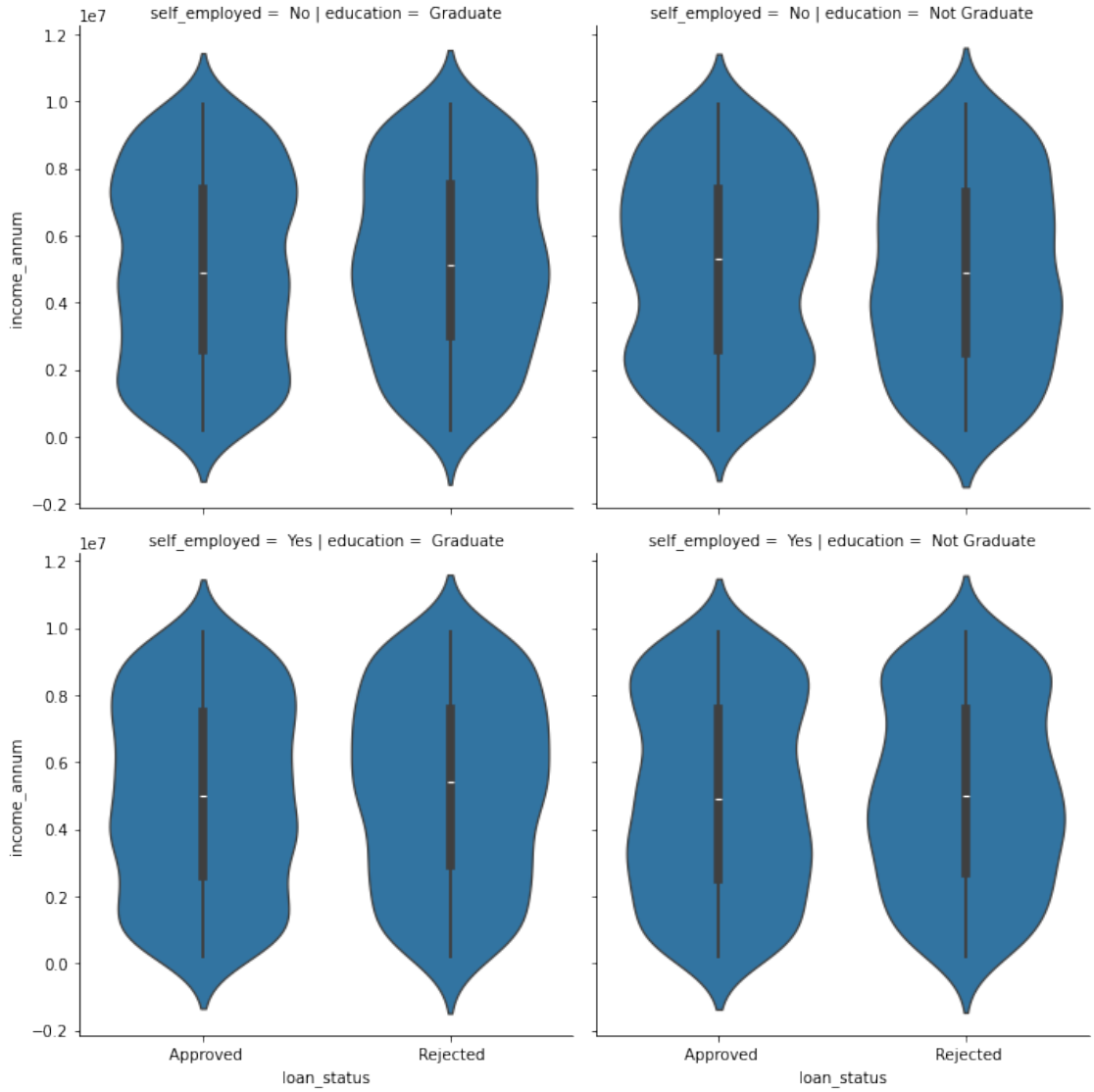
[27]:
```
sns.catplot(data=df,␣
↪x='loan_status',y='cibil_score',col='education',kind='box', row =␣
↪'self_employed')
```

[27]: <seaborn.axisgrid.FacetGrid at 0x1ff3357daf0>



[28]:
```
sns.catplot(data=df,␣
↪x='loan_status',y='income_annum',col='education',kind='violin', row =␣
↪'self_employed')
```

[28]: <seaborn.axisgrid.FacetGrid at 0x1ff3364ad60>



[29]:
```
# Data Preprocessing
```

[30]:
```
df.head()
```

[30]:

	loan_id	no_of_dependents	education	self_employed	income_annum	loan_amount	loan_term	cibil_score	residential_assets_value
0	1	2	Graduate	No	9600000	29900000	12	778	2400000
1	2	0	Not Graduate	Yes	4100000				
2	3	3	Graduate	No	9100000				
3	4	3	Graduate	No	8200000				
4	5	5	Not Graduate	Yes	9800000				

```
1     12200000          8          417                  2700000
2     29700000         20          506                  7100000
3     30700000          8          467                 18200000
4     24200000         20          382                 12400000

   commercial_assets_value  luxury_assets_value  bank_asset_value loan_status
0                 17600000             22700000           8000000    Approved
1                  2200000              8800000           3300000    Rejected
2                  4500000             33300000          12800000    Rejected
3                  3300000             23300000           7900000    Rejected
4                  8200000             29400000           5000000    Rejected
```

[31]:
```
df['education'].unique()
```

[31]:
```
array([' Graduate', ' Not Graduate'], dtype=object)
```

[32]:
```
df['education'] = df['education'].map({ ' Graduate':0, ' Not Graduate':1})
```

[33]:
```
df['loan_status'].unique()
```

[33]:
```
array([' Approved', ' Rejected'], dtype=object)
```

[34]:
```
df['loan_status'] = df['loan_status'].map({' Approved': 0, ' Rejected':1})
```

[35]:
```
df['self_employed'].unique()
```

[35]:
```
array([' No', ' Yes'], dtype=object)
```

[36]:
```
df['self_employed'] = df['self_employed'].map({' Yes':0,' No':1})
```

[37]:
```
df.head()
```

[37]:
```
   loan_id  no_of_dependents  education  self_employed  income_annum  \
0        1                 2          0              1       9600000
1        2                 0          1              0       4100000
2        3                 3          0              1       9100000
3        4                 3          0              1       8200000
4        5                 5          1              0       9800000

   loan_amount  loan_term  cibil_score  residential_assets_value  \
0     29900000         12          778                   2400000
1     12200000          8          417                   2700000
2     29700000         20          506                   7100000
3     30700000          8          467                  18200000
4     24200000         20          382                  12400000

   commercial_assets_value  luxury_assets_value  bank_asset_value  loan_status
```

```
0                17600000              22700000             8000000            0
1                 2200000               8800000             3300000            1
2                 4500000              33300000            12800000            1
3                 3300000              23300000             7900000            1
4                 8200000              29400000             5000000            1
```

[38]:
```
df = df.drop('loan_id', axis=1)
```

[39]:
```
df.head()
```

[39]:
```
   no_of_dependents  education  self_employed  income_annum  loan_amount  \
0                 2          0              1       9600000     29900000
1                 0          1              0       4100000     12200000
2                 3          0              1       9100000     29700000
3                 3          0              1       8200000     30700000
4                 5          1              0       9800000     24200000

   loan_term  cibil_score  residential_assets_value  commercial_assets_value  \
0         12          778                   2400000                 17600000
1          8          417                   2700000                  2200000
2         20          506                   7100000                  4500000
3          8          467                  18200000                  3300000
4         20          382                  12400000                  8200000

   luxury_assets_value  bank_asset_value  loan_status
0             22700000           8000000            0
1              8800000           3300000            1
2             33300000          12800000            1
3             23300000           7900000            1
4             29400000           5000000            1
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4269 entries, 0 to 4268
Data columns (total 12 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   no_of_dependents          4269 non-null   int64
 1   education                 4269 non-null   int64
 2   self_employed             4269 non-null   int64
 3   income_annum              4269 non-null   int64
 4   loan_amount               4269 non-null   int64
 5   loan_term                 4269 non-null   int64
 6   cibil_score               4269 non-null   int64
 7   residential_assets_value  4269 non-null   int64
 8   commercial_assets_value   4269 non-null   int64
```

```
 9   luxury_assets_value       4269 non-null   int64
 10  bank_asset_value          4269 non-null   int64
 11  loan_status               4269 non-null   int64
dtypes: int64(12)
memory usage: 400.3 KB
```

[41]:
```
# matrix plot
```

[42]:
```
corr = df.corr()
corr
```

[42]:

	no_of_dependents	education	self_employed	\
no_of_dependents	1.000000	-0.002697	-0.000765	
education	-0.002697	1.000000	-0.023224	
self_employed	-0.000765	-0.023224	1.000000	
income_annum	0.007266	-0.011625	-0.002368	
loan_amount	-0.003366	-0.010631	-0.001450	
loan_term	-0.020111	0.008417	-0.004107	
cibil_score	-0.009998	0.004649	0.004866	
residential_assets_value	0.007376	-0.010930	-0.006144	
commercial_assets_value	-0.001531	0.006763	0.017998	
luxury_assets_value	0.002817	-0.012471	-0.004413	
bank_asset_value	0.011163	-0.009424	0.000215	
loan_status	0.018114	0.004918	0.000345	

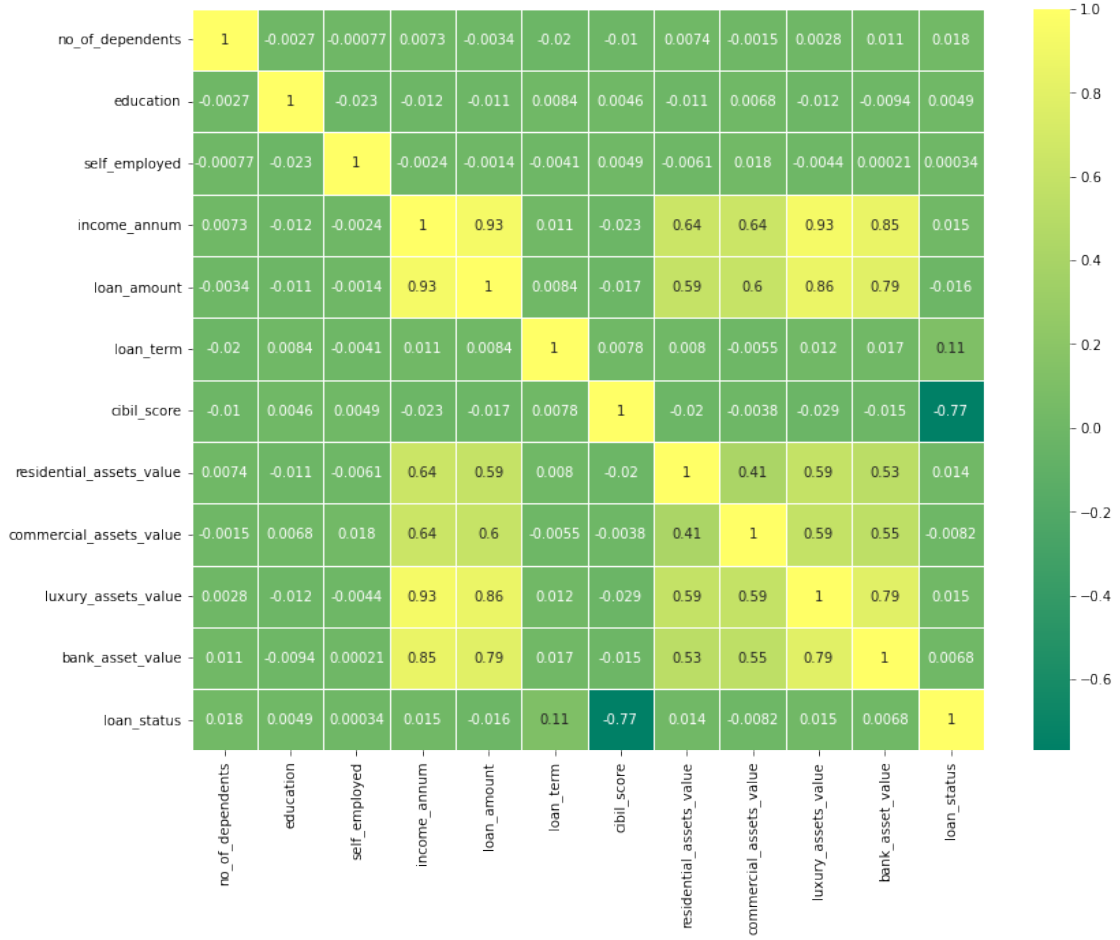
	income_annum	loan_amount	loan_term	cibil_score	\
no_of_dependents	0.007266	-0.003366	-0.020111	-0.009998	
education	-0.011625	-0.010631	0.008417	0.004649	
self_employed	-0.002368	-0.001450	-0.004107	0.004866	
income_annum	1.000000	0.927470	0.011488	-0.023034	
loan_amount	0.927470	1.000000	0.008437	-0.017035	
loan_term	0.011488	0.008437	1.000000	0.007810	
cibil_score	-0.023034	-0.017035	0.007810	1.000000	
residential_assets_value	0.636841	0.594596	0.008016	-0.019947	
commercial_assets_value	0.640328	0.603188	-0.005478	-0.003769	
luxury_assets_value	0.929145	0.860914	0.012490	-0.028618	
bank_asset_value	0.851093	0.788122	0.017177	-0.015478	
loan_status	0.015189	-0.016150	0.113036	-0.770518	

	residential_assets_value	commercial_assets_value	\
no_of_dependents	0.007376	-0.001531	
education	-0.010930	0.006763	
self_employed	-0.006144	0.017998	
income_annum	0.636841	0.640328	
loan_amount	0.594596	0.603188	
loan_term	0.008016	-0.005478	
cibil_score	-0.019947	-0.003769	

```
residential_assets_value                  1.000000                0.414786
commercial_assets_value                   0.414786                1.000000
luxury_assets_value                       0.590932                0.591128
bank_asset_value                          0.527418                0.548576
loan_status                               0.014367               -0.008246

                          luxury_assets_value  bank_asset_value  loan_status
no_of_dependents                     0.002817          0.011163     0.018114
education                           -0.012471         -0.009424     0.004918
self_employed                       -0.004413          0.000215     0.000345
income_annum                         0.929145          0.851093     0.015189
loan_amount                          0.860914          0.788122    -0.016150
loan_term                            0.012490          0.017177     0.113036
cibil_score                         -0.028618         -0.015478    -0.770518
residential_assets_value             0.590932          0.527418     0.014367
commercial_assets_value              0.591128          0.548576    -0.008246
luxury_assets_value                  1.000000          0.788517     0.015465
bank_asset_value                     0.788517          1.000000     0.006778
loan_status                          0.015465          0.006778     1.000000
```

[43]:
```
plt.figure(figsize = (13,10))
sns.heatmap(corr, annot = True, cmap ='summer', linewidth = 0.5)
```

[43]: <Axes: >



[51]:
```
# train - test split
```

[52]:
```
from sklearn.model_selection import train_test_split
```

[53]:
```
x = df.drop('loan_status', axis=1)
```

[56]:
```
y = df['loan_status']
```

[58]:
```
x.head()
```

[58]:

	no_of_dependents	education	self_employed	income_annum	loan_amount
0	2	0	1	9600000	29900000
1	0	1	0	4100000	12200000
2	3	0	1	9100000	29700000
3	3	0	1	8200000	30700000
4	5	1	0	9800000	24200000

	loan_term	cibil_score	residential_assets_value	commercial_assets_value

```
0          12          778                2400000                17600000
1           8          417                2700000                 2200000
2          20          506                7100000                 4500000
3           8          467               18200000                 3300000
4          20          382               12400000                 8200000

   luxury_assets_value  bank_asset_value
0             22700000           8000000
1              8800000           3300000
2             33300000          12800000
3             23300000           7900000
4             29400000           5000000
```

[59]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y, random_state = 42,␣
↪test_size = 0.3)
```

[60]:
```
# Before using logistic regression model we should do standardization
```

[61]:
```
from sklearn.preprocessing import StandardScaler
scaler  = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

[84]:
```
from sklearn.linear_model import LogisticRegression
log = LogisticRegression()
log.fit(x_train_scaled, y_train)
y_pred = log.predict(x_test_scaled)
```

[85]:
```
from sklearn.metrics import accuracy_score, r2_score, confusion_matrix
```

[86]:
```
ac = accuracy_score(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
cm  = confusion_matrix (y_test, y_pred)
print(ac, r2)
```

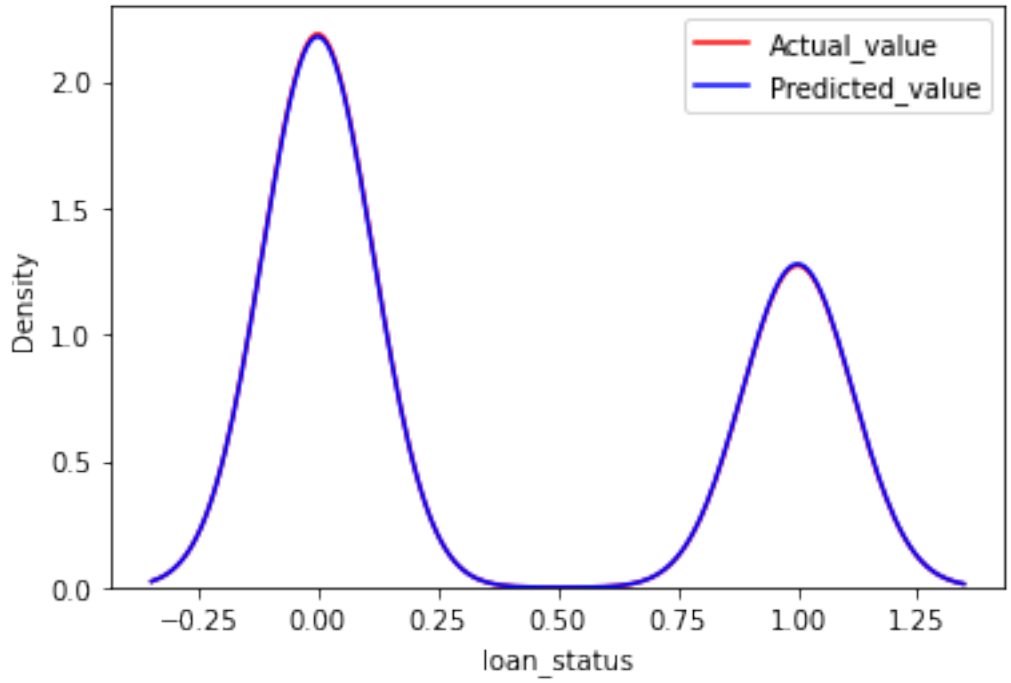
```
0.9039812646370023 0.5870016513328615
```

[87]:
```
print(cm)
```

```
[[747  63]
 [ 60 411]]
```

[88]:
```
sns.distplot(y_test,color='red', hist = False, label = 'Actual_value')
sns.distplot(y_pred, color='blue', hist = False, label = 'Predicted_value')
plt.legend()
plt.show()
```

```
C:\Users\amits\AppData\Local\Temp/ipykernel_13400/4052658849.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(y_test,color='red', hist = False, label = 'Actual_value')
C:\Users\amits\AppData\Local\Temp/ipykernel_13400/4052658849.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(y_pred, color='blue', hist = False, label = 'Predicted_value')
```



[89]:
```
# Random Forest Classifier
```

[97]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
rf = RandomForestClassifier()


param_grid = {
    'n_estimators' : [100,200,300,400,600,800,1000],
    'criterion' :['gini', 'entropy'],
    'max_depth': [5,10,14,18,20],
    'min_samples_split' : [2,4,8,12,16,20],
    'min_samples_leaf' : [2,5,8,10,15,20]

}

grid_search = GridSearchCV(rf, param_grid, verbose = 1, n_jobs = -1, cv=5)

grid_search.fit(x_train, y_train)
grid_search.best_params_
```

```
Fitting 5 folds for each of 2520 candidates, totalling 12600 fits
```

[97]:
```
{'criterion': 'entropy',
 'max_depth': 20,
 'min_samples_leaf': 2,
 'min_samples_split': 2,
 'n_estimators': 200}
```

[99]:
```
grid_search.best_estimator_
```

[99]:
```
RandomForestClassifier(criterion='entropy', max_depth=20, min_samples_leaf=2,
                       n_estimators=200)
```

[100]:
```
rfc = RandomForestClassifier(criterion='entropy', max_depth=20,␣
 ↪min_samples_leaf=2,
                       n_estimators=200)
rfc.fit(x_train, y_train)
y_pred = rfc.predict(x_test)
```

[101]:
```
ac = accuracy_score(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
cm  = confusion_matrix (y_test, y_pred)
print(ac, r2)
```

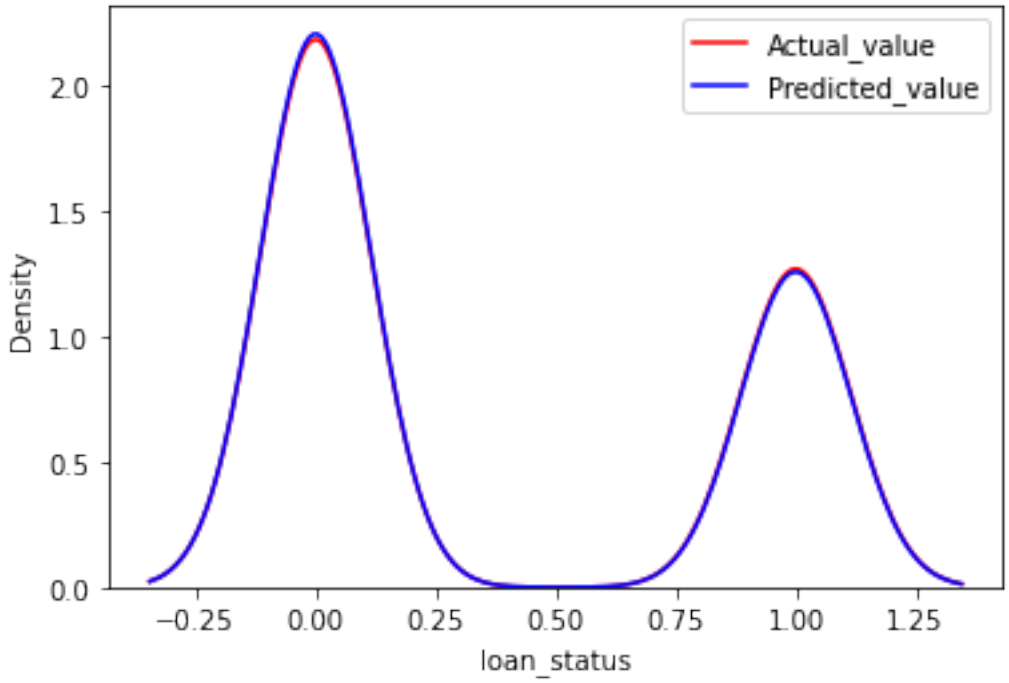
```
0.9781420765027322 0.9059841157505701
```

[102]:
```
print(cm)
```

```
[[799  11]
 [ 17 454]]
```

[103]:
```
sns.distplot(y_test,color='red', hist = False, label = 'Actual_value')
sns.distplot(y_pred, color='blue', hist = False, label = 'Predicted_value')
plt.legend()
plt.show()
```

```
C:\Users\amits\AppData\Local\Temp/ipykernel_13400/4052658849.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(y_test,color='red', hist = False, label = 'Actual_value')
C:\Users\amits\AppData\Local\Temp/ipykernel_13400/4052658849.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(y_pred, color='blue', hist = False, label = 'Predicted_value')
```



[ ]:
```
import pickle

pickle.dump(rfc, open())
```