In [2]:

```
import os
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline
from sklearn.model_selection import train_test_split
```

In [3]:

```
os.chdir('C:\\Excel Class')
```

In [4]:

```
df=pd.read_csv('loan_data.csv')
df
```

Out[4]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with.cr
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.95
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.00
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4710.00
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2699.95
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	4066.00
...	...	...	...	...	...	...	...	
9573	0	all_other	0.1461	344.76	12.180755	10.39	672	10474.00
9574	0	all_other	0.1253	257.70	11.141862	0.21	722	4380.00
9575	0	debt_consolidation	0.1071	97.81	10.596635	13.09	687	3450.04
9576	0	home_improvement	0.1600	351.58	10.819778	19.18	692	1800.00
9577	0	debt_consolidation	0.1392	853.43	11.264464	16.28	732	4740.00

9578 rows × 14 columns

In [5]:

```
df.describe()
```

Out[5]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.c
count	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.00
mean	0.804970	0.122640	319.089413	10.932117	12.606679	710.846314	4560.76
std	0.396245	0.026847	207.071301	0.614813	6.883970	37.970537	2496.93
min	0.000000	0.060000	15.670000	7.547502	0.000000	612.000000	178.95
25%	1.000000	0.103900	163.770000	10.558414	7.212500	682.000000	2820.00
50%	1.000000	0.122100	268.950000	10.928884	12.665000	707.000000	4139.95
75%	1.000000	0.140700	432.762500	11.291293	17.950000	737.000000	5730.00
max	1.000000	0.216400	940.140000	14.528354	29.960000	827.000000	17639.95

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   credit.policy      9578 non-null   int64  
 1   purpose            9578 non-null   object 
 2   int.rate           9578 non-null   float64
 3   installment        9578 non-null   float64
 4   log.annual.inc     9578 non-null   float64
 5   dti                9578 non-null   float64
 6   fico               9578 non-null   int64  
 7   days.with.cr.line  9578 non-null   float64
 8   revol.bal          9578 non-null   int64  
 9   revol.util         9578 non-null   float64
 10  inq.last.6mths     9578 non-null   int64  
 11  delinq.2yrs        9578 non-null   int64  
 12  pub.rec            9578 non-null   int64  
 13  not.fully.paid     9578 non-null   int64  
dtypes: float64(6), int64(7), object(1)
memory usage: 1.0+ MB
```

In [7]:

```
df.isnull().sum()
```

Out[7]:

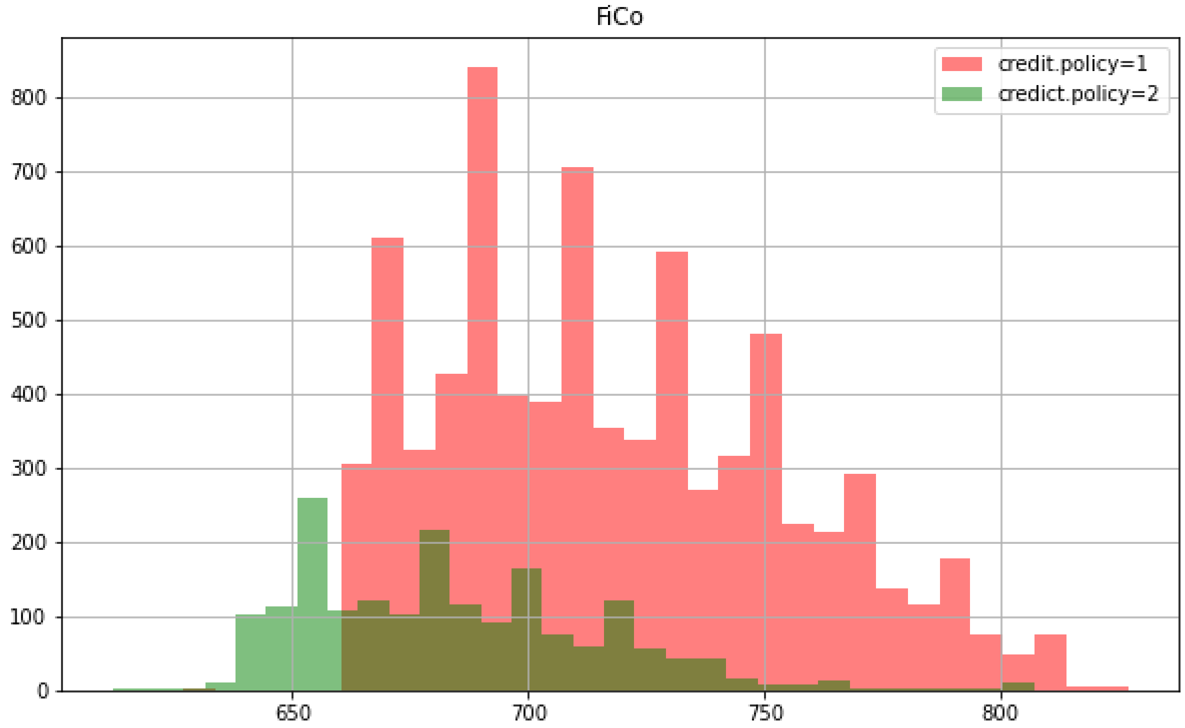
```
credit.policy        0
purpose              0
int.rate             0
installment          0
log.annual.inc       0
dti                  0
fico                 0
days.with.cr.line    0
revol.bal            0
revol.util           0
inq.last.6mths       0
delinq.2yrs          0
pub.rec              0
not.fully.paid       0
dtype: int64
```

In [10]:

```
plt.figure(figsize=(10,6))
df[df['credit.policy']==1]['fico'].hist(alpha=0.5,color='red',bins=30,label='credit.
df[df['credit.policy']==0]['fico'].hist(alpha=0.5,color='green',bins=30,label='credi
plt.legend()
plt.title('FiCo')
```

Out[10]:

```
Text(0.5, 1.0, 'FiCo')
```



In [12]:

```
df['credit.policy']
```

Out[12]:

```
0       1
1       1
2       1
3       1
4       1
       ..
9573    0
9574    0
9575    0
9576    0
9577    0
Name: credit.policy, Length: 9578, dtype: int64
```

In [13]:

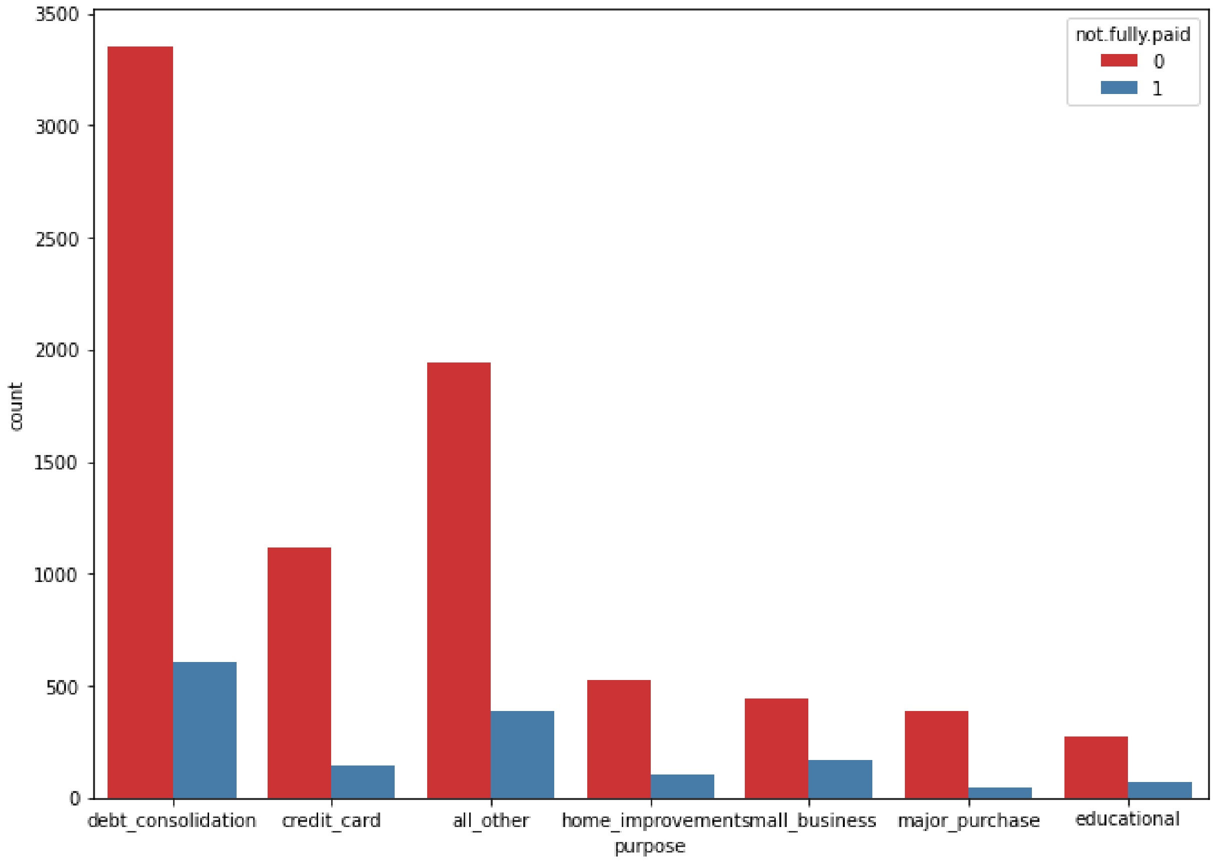
```
import seaborn as sns
```

In [18]:

```
plt.figure(figsize=(11,8))
sns.countplot(x='purpose',hue='not.fully.paid',data=df,palette='Set1')
```

Out[18]:

```
<AxesSubplot:xlabel='purpose', ylabel='count'>
```



In [20]:

```
df.dtypes
```

Out[20]:

```
credit.policy          int64
purpose               object
int.rate             float64
installment          float64
log.annual.inc       float64
dti                  float64
fico                   int64
days.with.cr.line    float64
revol.bal              int64
revol.util           float64
inq.last.6mths         int64
delinq.2yrs            int64
pub.rec                int64
not.fully.paid         int64
dtype: object
```

In [23]:

```
cat = ['purpose']
cat
```

Out[23]:

```
['purpose']
```

In [24]:

```
final_data=pd.get_dummies(df,columns=cat,drop_first=True)
final_data
```

Out[24]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revc
0	1	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	
1	1	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	
2	1	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revc
3	1	0.1008	162.34	11.350407	8.10	712	2699.958333	33667	
4	1	0.1426	102.92	11.299732	14.97	667	4066.000000	4740	
...	...	...	...	...	...	...	...	...	
9573	0	0.1461	344.76	12.180755	10.39	672	10474.000000	215372	
9574	0	0.1253	257.70	11.141862	0.21	722	4380.000000	184	
9575	0	0.1071	97.81	10.596635	13.09	687	3450.041667	10036	
9576	0	0.1600	351.58	10.819778	19.18	692	1800.000000	0	
9577	0	0.1392	853.43	11.264464	16.28	732	4740.000000	37879	

9578 rows × 19 columns

In [25]:

```
final_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 19 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   credit.policy               9578 non-null   int64  
 1   int.rate                    9578 non-null   float64
 2   installment                 9578 non-null   float64
 3   log.annual.inc              9578 non-null   float64
 4   dti                         9578 non-null   float64
 5   fico                        9578 non-null   int64  
 6   days.with.cr.line           9578 non-null   float64
 7   revol.bal                   9578 non-null   int64  
 8   revol.util                  9578 non-null   float64
 9   inq.last.6mths              9578 non-null   int64  
 10  delinq.2yrs                 9578 non-null   int64  
 11  pub.rec                     9578 non-null   int64  
 12  not.fully.paid              9578 non-null   int64  
 13  purpose_credit_card         9578 non-null   uint8  
 14  purpose_debt_consolidation  9578 non-null   uint8  
 15  purpose_educational         9578 non-null   uint8  
 16  purpose_home_improvement    9578 non-null   uint8  
 17  purpose_major_purchase      9578 non-null   uint8  
 18  purpose_small_business      9578 non-null   uint8  
dtypes: float64(6), int64(7), uint8(6)
memory usage: 1.0 MB
```

In [26]:

```
x=final_data.drop('not.fully.paid',axis=1)
x
```

Out[26]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revc
0	1	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	
1	1	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	
2	1	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	
3	1	0.1008	162.34	11.350407	8.10	712	2699.958333	33667	
4	1	0.1426	102.92	11.299732	14.97	667	4066.000000	4740	
...	...	...	...	...	...	...	...	...	

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revc
9573	0	0.1461	344.76	12.180755	10.39	672	10474.000000	215372	
9574	0	0.1253	257.70	11.141862	0.21	722	4380.000000	184	
9575	0	0.1071	97.81	10.596635	13.09	687	3450.041667	10036	
9576	0	0.1600	351.58	10.819778	19.18	692	1800.000000	0	
9577	0	0.1392	853.43	11.264464	16.28	732	4740.000000	37879	

9578 rows × 18 columns

In [27]:

```
y=final_data['not.fully.paid']
y
```

Out[27]:

```
0       0
1       0
2       0
3       0
4       0
       ..
9573    1
9574    1
9575    1
9576    1
9577    1
Name: not.fully.paid, Length: 9578, dtype: int64
```

In [28]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

In [29]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [30]:

```
dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
```

Out[30]:

```
DecisionTreeClassifier()
```

In [31]:

```
pred=dt.predict(x_test)
pred
```

Out[31]:

```
array([1, 1, 0, ..., 0, 0, 0], dtype=int64)
```

In [40]:

```
from sklearn.metrics import accuracy_score,classification_report
print(accuracy_score(y_test,pred))
print(classification_report(y_test,pred))
```

```
0.7458246346555324
              precision    recall  f1-score   support

           0       0.85      0.84      0.85      1593
           1       0.25      0.26      0.25       323

    accuracy                           0.75      1916
   macro avg       0.55      0.55      0.55      1916
```

```
weighted avg       0.75      0.75      0.75      1916
```

In [34]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [35]:

```
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
```

Out[35]:

```
RandomForestClassifier()
```

In [36]:

```
predy=rf.predict(x_test)
predy
```

Out[36]:

```
array([0, 1, 0, ..., 0, 0, 0], dtype=int64)
```

In [41]:

```
print(accuracy_score(y_test,predy))
print(classification_report(y_test,predy))
```

```
0.8335073068893528
              precision    recall  f1-score   support

           0       0.84      1.00      0.91      1593
           1       0.61      0.03      0.06       323

    accuracy                           0.83      1916
   macro avg       0.72      0.51      0.49      1916
weighted avg       0.80      0.83      0.77      1916
```

In [ ]:

In [ ]: