```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

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

```
df.head()
```

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fic
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737
1	1	credit_card	0.1071	228.22	11.082143	14.29	707
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712
4	1	credit_card	0.1426	102.92	11.299732	14.97	667

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

```
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
```

```
df.dtypes
```

```
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
```

```
###  df=df.drop(['purpose'], axis=1)
```

```
df.head()
```

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.lin
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.95833
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.00000
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4710.00000
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2699.95833
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	4066.00000

```
df.describe()
```

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.l
count	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000
mean	0.804970	0.122640	319.089413	10.932117	12.606679	710.846314	4560.767
std	0.396245	0.026847	207.071301	0.614813	6.883970	37.970537	2496.930
min	0.000000	0.060000	15.670000	7.547502	0.000000	612.000000	178.958
25%	1.000000	0.103900	163.770000	10.558414	7.212500	682.000000	2820.000
50%	1.000000	0.122100	268.950000	10.928884	12.665000	707.000000	4139.958
75%	1.000000	0.140700	432.762500	11.291293	17.950000	737.000000	5730.000

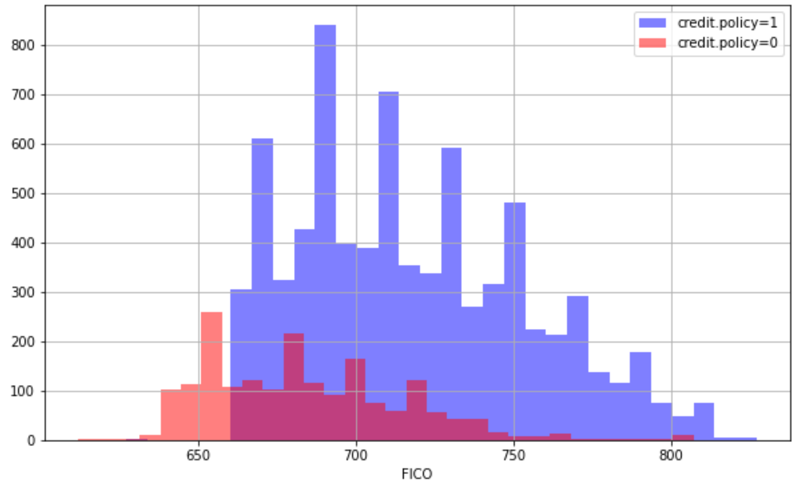
```
df["credit.policy"].value_counts()
```

```
1    7710
0    1868
Name: credit.policy, dtype: int64
```

```
import seaborn as sns
```

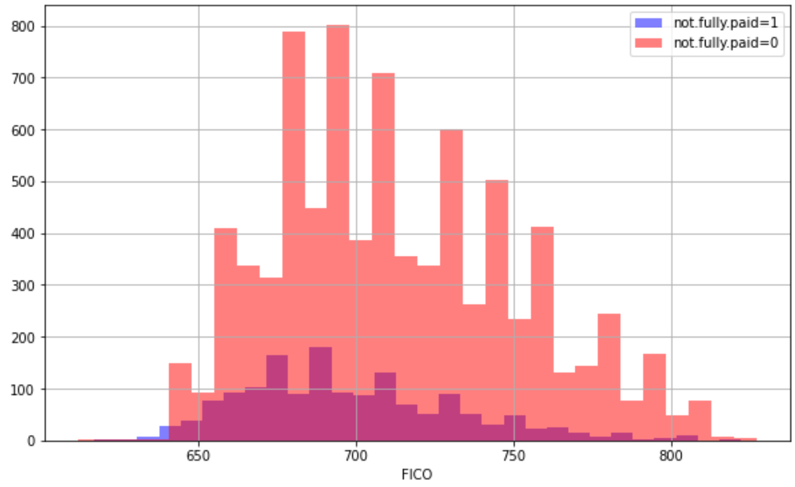
```
plt.figure(figsize=(10,6))
df[df['credit.policy']==1]['fico'].hist(alpha=0.5, color='blue', bins=30, label='credit.policy=1')
df[df['credit.policy']==0]['fico'].hist(alpha=0.5, color='red', bins=30, label='credit.policy=0')
plt.legend()
plt.xlabel('FICO')
```

```
Text(0.5, 0, 'FICO')
```



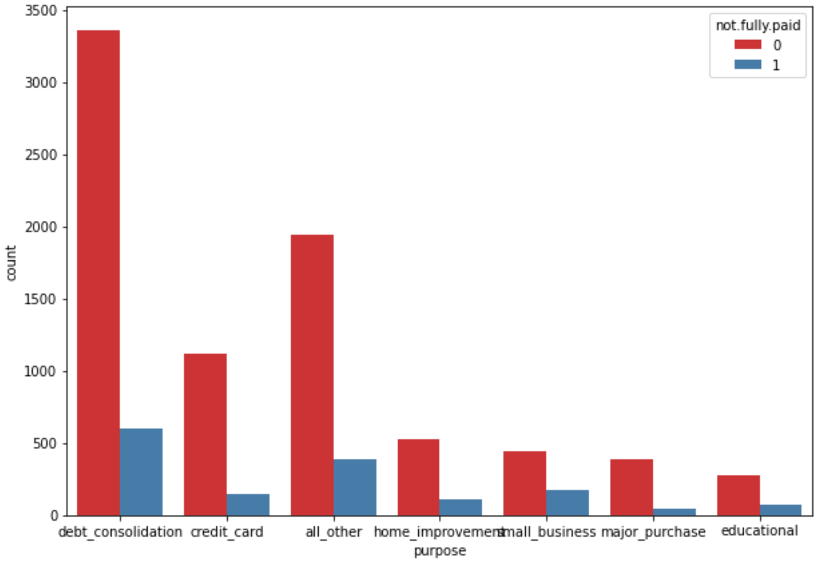
```
plt.figure(figsize=(10,6))
df[df['not.fully.paid']==1]['fico'].hist(alpha=0.5, color='blue', bins=30, label='not.fully.paid=1')
df[df['not.fully.paid']==0]['fico'].hist(alpha=0.5, color='red', bins=30, label='not.fully.paid=0')
plt.legend()
plt.xlabel('FICO')
```

```
Text(0.5, 0, 'FICO')
```



```
plt.figure(figsize=(10,7))
sns.countplot(x='purpose', hue='not.fully.paid', data=df, palette='Set1')
```

```
    <matplotlib.axes._subplots.AxesSubplot at 0x7f87aaa5fd90>
```



```
df=df.drop(['purpose'], axis=1)
```

```
X=df.drop('not.fully.paid', axis=1)
y=df['not.fully.paid']
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state= 50)
```

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

```
dt=DecisionTreeClassifier()
```

```
dt.fit(X_train, y_train)
```

```
    DecisionTreeClassifier()
```

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

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
print(classification_report(y_test, pred))
```

```
                  precision    recall  f1-score   support

               0       0.86      0.81      0.84      1620
               1       0.21      0.27      0.23       296

        accuracy                           0.73      1916
       macro avg       0.53      0.54      0.54      1916
    weighted avg       0.76      0.73      0.74      1916
```

```
print(confusion_matrix(y_test, pred))
```

```
    [[1320  300]
     [ 217   79]]
```

```
print(accuracy_score(y_test, pred)*100)
```

```
    73.01670146137788
```

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

```
rf = RandomForestClassifier()
```

```
rf.fit(X_train, y_train)
```

```
    RandomForestClassifier()
```

```
prediction=rf.predict(X_test)
```

```
print(classification_report(y_test, prediction))
```

```
              precision    recall  f1-score   support

           0       0.85      0.99      0.92      1620
           1       0.48      0.03      0.06       296

    accuracy                           0.84      1916
   macro avg       0.66      0.51      0.49      1916
weighted avg       0.79      0.84      0.78      1916
```

```
print(confusion_matrix(y_test, prediction))
```

```
[[1609   11]
 [ 286   10]]
```

```
print(accuracy_score(y_test, prediction)*100)
```

```
84.49895615866389
```