In [251]:

```
#importing the necessary libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [252]:

```
#reading the dataset

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

Out[252]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4710.
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2699.
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	4066.
...	...	...	...	...	...	...	...	
9573	0	all_other	0.1461	344.76	12.180755	10.39	672	10474.
9574	0	all_other	0.1253	257.70	11.141862	0.21	722	4380.
9575	0	debt_consolidation	0.1071	97.81	10.596635	13.09	687	3450.
9576	0	home_improvement	0.1600	351.58	10.819778	19.18	692	1800.
9577	0	debt_consolidation	0.1392	853.43	11.264464	16.28	732	4740.

9578 rows × 14 columns

In [253]:

```
#checking for null values

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

Out[253]:

```
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 [254]:

```
#by having a look at the dataset we can say that 'not.fully.paid' is the output variable
#as always it is our duty to make sure that the data is not imbalanced.

df['not.fully.paid'].value_counts()
```

Out[254]:

```
0    8045
1    1533
Name: not.fully.paid, dtype: int64
```

In [255]:

```
#as we can see that the data is imbalanced.
#so we will treat this by oversampling our minority class by using RandomOverSampler() func
```

Treating the imbalanced data

In [79]:

```
import imblearn
from imblearn.over_sampling import RandomOverSampler
```

In [256]:

```
#oversampling the minority class.

ros = RandomOverSampler(sampling_strategy={1:4000})
a = ros.fit_resample(df , df['not.fully.paid'])
df1=a[0]
df1
```

Out[256]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	2
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	4
...	...	...	...	...	...	...	...	
12040	1	all_other	0.1322	169.01	10.645425	13.49	717	3
12041	1	debt_consolidation	0.1253	669.33	11.350407	24.56	717	4
12042	1	home_improvement	0.1670	710.03	11.264464	24.60	677	1
12043	1	major_purchase	0.1095	98.15	10.980263	7.40	722	5
12044	0	debt_consolidation	0.1474	690.74	12.016098	12.38	682	3

12045 rows × 14 columns

In [257]:

```
#let's have a look at our dataset.

df1['not.fully.paid'].value_counts()
```

Out[257]:

```
0    8045
1    4000
Name: not.fully.paid, dtype: int64
```

In [258]:

```
#as mentioned in the project guidelines , we will now turn our categories variable(string)
#in other words we will one hot encode our df['purpose'] column.

from sklearn.preprocessing import OneHotEncoder
```

In [259]:

```
oe = OneHotEncoder()
encoding=oe.fit_transform(np.array(df1['purpose']).reshape(-1,1))
encoding_df = pd.DataFrame(encoding.toarray() , columns=df1['purpose'].value_counts().index
encoding_df
```

Out[259]:

	debt_consolidation	all_other	credit_card	small_business	home_improvement	major_
0	0.0	0.0	1.0	0.0	0.0	
1	0.0	1.0	0.0	0.0	0.0	
2	0.0	0.0	1.0	0.0	0.0	
3	0.0	0.0	1.0	0.0	0.0	
4	0.0	1.0	0.0	0.0	0.0	
...	...	...	...	...	...	
12040	1.0	0.0	0.0	0.0	0.0	
12041	0.0	0.0	1.0	0.0	0.0	
12042	0.0	0.0	0.0	0.0	1.0	
12043	0.0	0.0	0.0	0.0	0.0	
12044	0.0	0.0	1.0	0.0	0.0	

12045 rows × 7 columns

In [260]:

```
#now just combine our encodings with original df.

df2 = pd.concat((df1 , encoding_df) , axis=1)
df2
```

Out[260]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.wi
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	563
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	276
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	471
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	269
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	406
...	...	...	...	...	...	...	...	
12040	1	all_other	0.1322	169.01	10.645425	13.49	717	333
12041	1	debt_consolidation	0.1253	669.33	11.350407	24.56	717	483
12042	1	home_improvement	0.1670	710.03	11.264464	24.60	677	174
12043	1	major_purchase	0.1095	98.15	10.980263	7.40	722	504
12044	0	debt_consolidation	0.1474	690.74	12.016098	12.38	682	399

12045 rows × 21 columns

In [261]:

```
#as we know that df['not.fully.paid'] is our target variable , so i prefer to keep it in th
#so , i am first deleting the column df['df.fully.paid'] and than adding it again , so that

df2 = df2.drop('not.fully.paid' , axis=1)
df2 = pd.concat((df2 , df1['not.fully.paid']) , axis=1)
df2
```

Out[261]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.wi
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	563
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	276
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	471
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	269
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	406
...	...	...	...	...	...	...	...	
12040	1	all_other	0.1322	169.01	10.645425	13.49	717	333
12041	1	debt_consolidation	0.1253	669.33	11.350407	24.56	717	483
12042	1	home_improvement	0.1670	710.03	11.264464	24.60	677	174
12043	1	major_purchase	0.1095	98.15	10.980263	7.40	722	504
12044	0	debt_consolidation	0.1474	690.74	12.016098	12.38	682	399

12045 rows × 21 columns

In [262]:

```
#let's check the the correlation of our feature columns with target variable

df2.corrwith(df2['not.fully.paid'])
```

Out[262]:

```
credit.policy        -0.194118
int.rate              0.210788
installment           0.062471
log.annual.inc       -0.044501
dti                   0.051914
fico                 -0.196458
days.with.cr.line    -0.035792
revol.bal             0.049912
revol.util            0.109987
inq.last.6mths        0.183427
delinq.2yrs           0.013808
pub.rec               0.057448
debt_consolidation    0.008888
all_other            -0.066946
credit_card          -0.018338
small_business        0.030573
home_improvement      0.005441
major_purchase       -0.035901
educational           0.103722
not.fully.paid        1.000000
dtype: float64
```

Data Visualisation

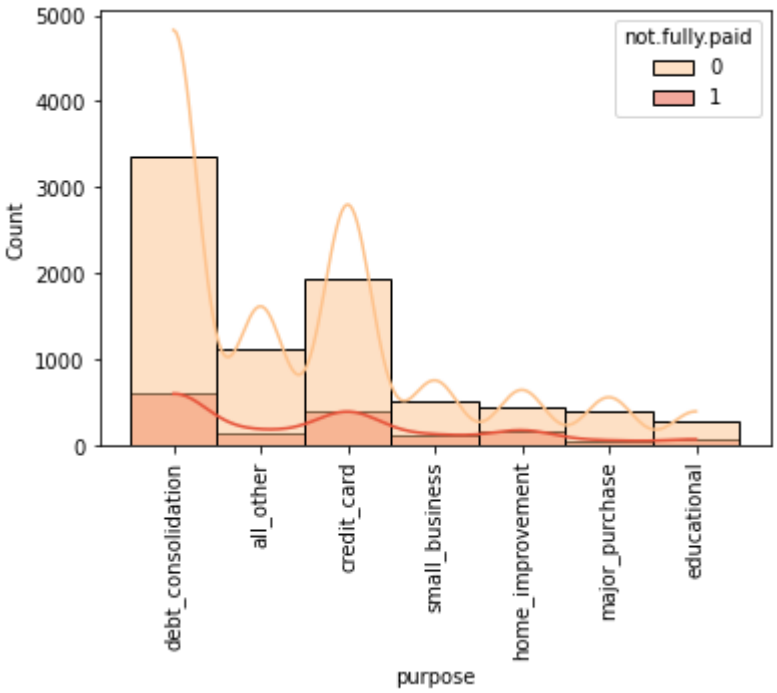
In [163]:

```
g = sns.histplot(x=df['purpose'] , hue=df['not.fully.paid'] ,cbar=True , kde=True ,palette=
g.set_xticklabels(labels=df2['purpose'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\1151361187.py:2: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['purpose'].value_counts().index,rotation=90)
```

Out[163]:

```
[Text(0, 0, 'debt_consolidation'),
 Text(1, 0, 'all_other'),
 Text(2, 0, 'credit_card'),
 Text(3, 0, 'small_business'),
 Text(4, 0, 'home_improvement'),
 Text(5, 0, 'major_purchase'),
 Text(6, 0, 'educational')]
```



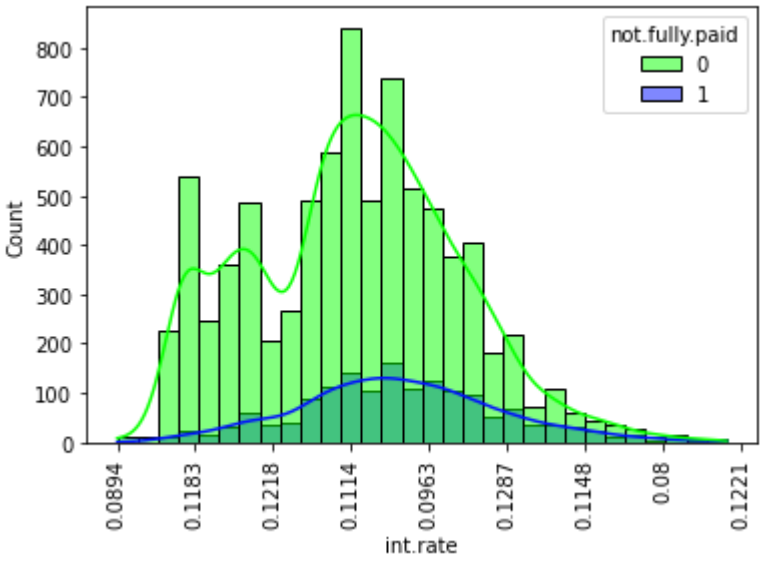
In [178]:

```
g = sns.histplot(x=df['int.rate'] , hue=df['not.fully.paid'] ,bins=30 , cbar=True , kde=Tru
g.set_xticklabels(labels=df2['int.rate'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\1430898597.py:2: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['int.rate'].value_counts().index,rotation=90)
```

Out[178]:

```
[Text(0.04, 0, '0.1253'),
 Text(0.06, 0, '0.0894'),
 Text(0.08, 0, '0.1183'),
 Text(0.1, 0, '0.1218'),
 Text(0.12, 0, '0.1114'),
 Text(0.14, 0, '0.0963'),
 Text(0.16, 0, '0.1287'),
 Text(0.18000000000000002, 0, '0.1148'),
 Text(0.2, 0, '0.08'),
 Text(0.22, 0, '0.1221'),
 Text(0.24000000000000002, 0, '0.0859')]
```



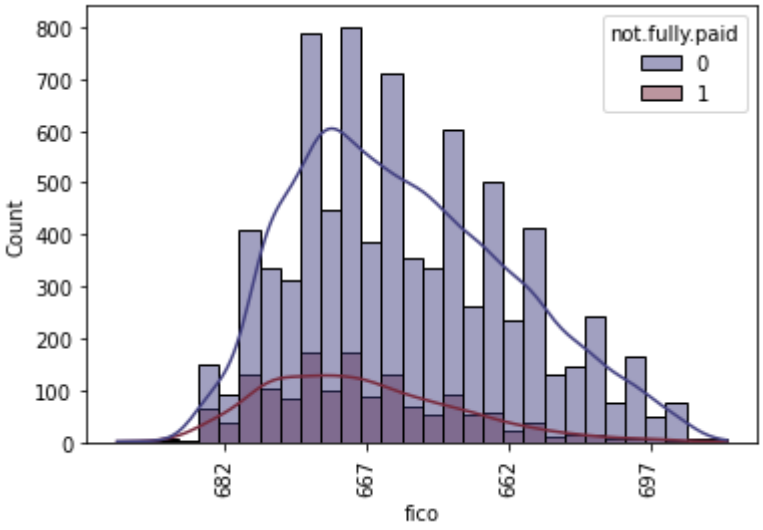
In [177]:

```
g = sns.histplot(x=df["fico"] , hue=df['not.fully.paid'] ,bins=30,cbar=True , kde=True ,pal
g.set_xticklabels(labels=df2['fico'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\2147369090.py:2: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['fico'].value_counts().index,rotation=90)
```

Out[177]:

```
[Text(600.0, 0, '687'),
 Text(650.0, 0, '682'),
 Text(700.0, 0, '667'),
 Text(750.0, 0, '662'),
 Text(800.0, 0, '697'),
 Text(850.0, 0, '692')]
```



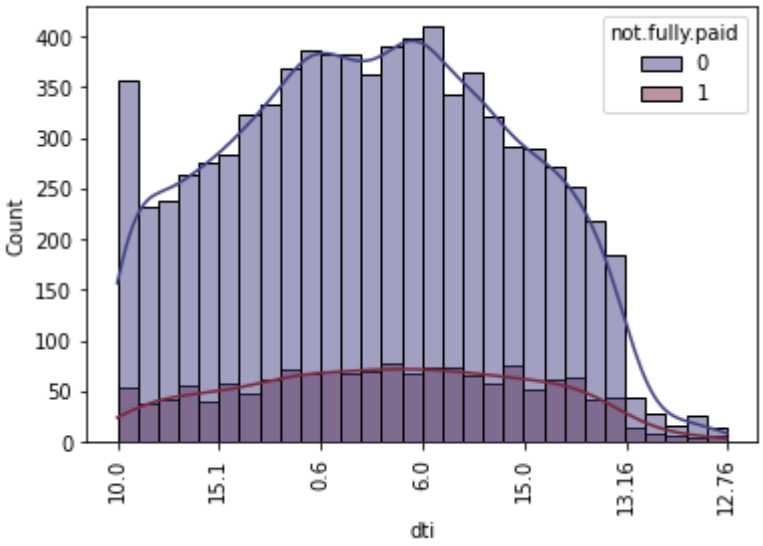
In [176]:

```
g = sns.histplot(x=df["dti"] , hue=df['not.fully.paid'] ,bins=30,cbar=True , kde=True ,pale
g.set_xticklabels(labels=df2['dti'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\2872073709.py:2: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['dti'].value_counts().index,rotation=90)
```

Out[176]:

```
[Text(-5.0, 0, '0.0'),
 Text(0.0, 0, '10.0'),
 Text(5.0, 0, '15.1'),
 Text(10.0, 0, '0.6'),
 Text(15.0, 0, '6.0'),
 Text(20.0, 0, '15.0'),
 Text(25.0, 0, '13.16'),
 Text(30.0, 0, '12.76'),
 Text(35.0, 0, '20.37')]
```



In [ ]:

```
g = sns.histplot(x=df[""] , hue=df['not.fully.paid'] ,bins=30,cbar=True , kde=True ,palette
g.set_xticklabels(labels=df2['purpose'].value_counts().index,rotation=90)
```

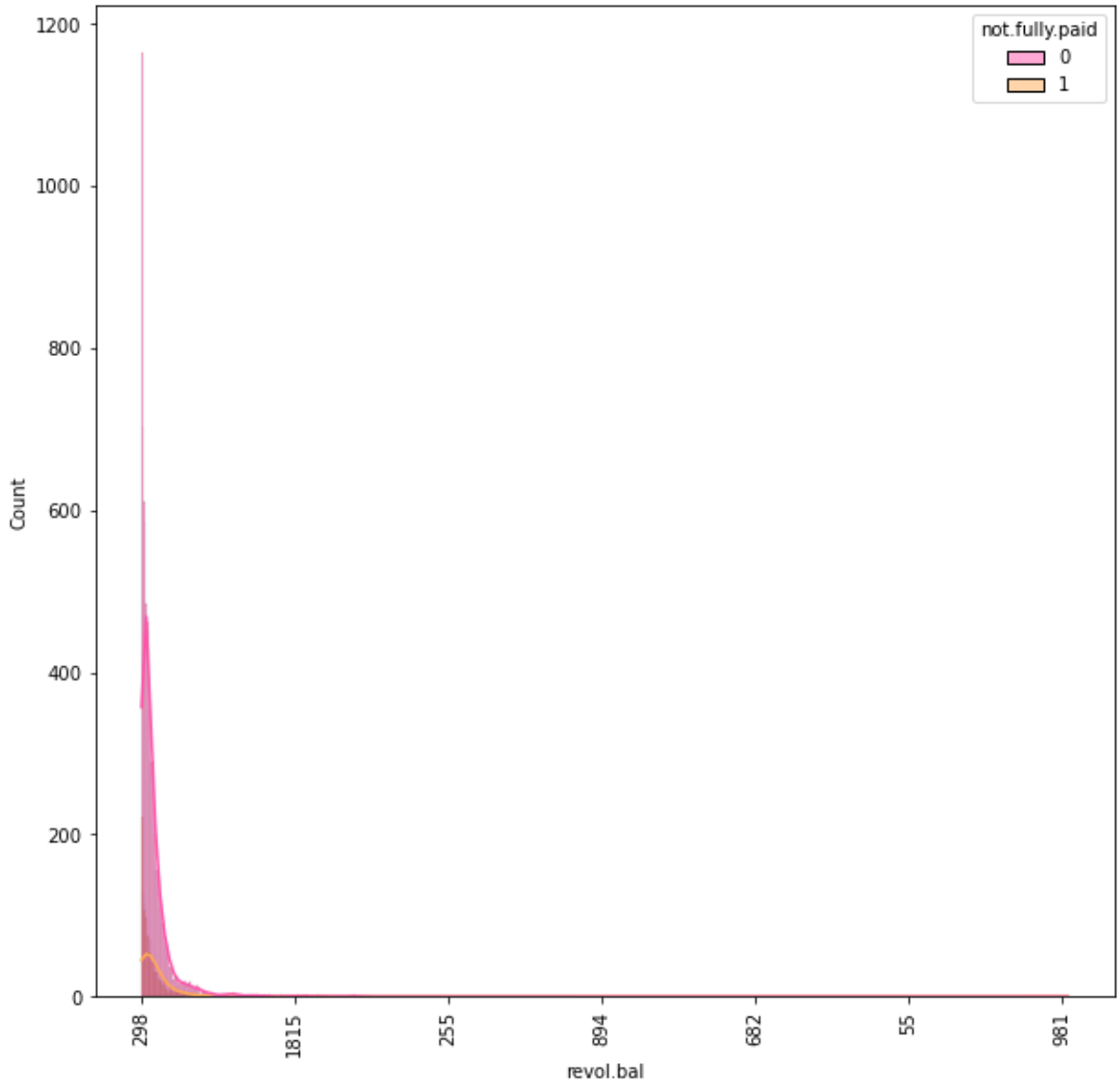
In [192]:

```
plt.figure(figsize=(10,10))
g = sns.histplot(x=df["revol.bal"] , hue=df['not.fully.paid'] ,cbar=True , kde=True ,palett
g.set_xticklabels(labels=df2['revol.bal'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\4131124766.py:3: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['revol.bal'].value_counts().index,rotation=9
0)
```

Out[192]:

```
[Text(-200000.0, 0, '0'),
 Text(0.0, 0, '298'),
 Text(200000.0, 0, '1815'),
 Text(400000.0, 0, '255'),
 Text(600000.0, 0, '894'),
 Text(800000.0, 0, '682'),
 Text(1000000.0, 0, '55'),
 Text(1200000.0, 0, '981'),
 Text(1400000.0, 0, '2229')]
```



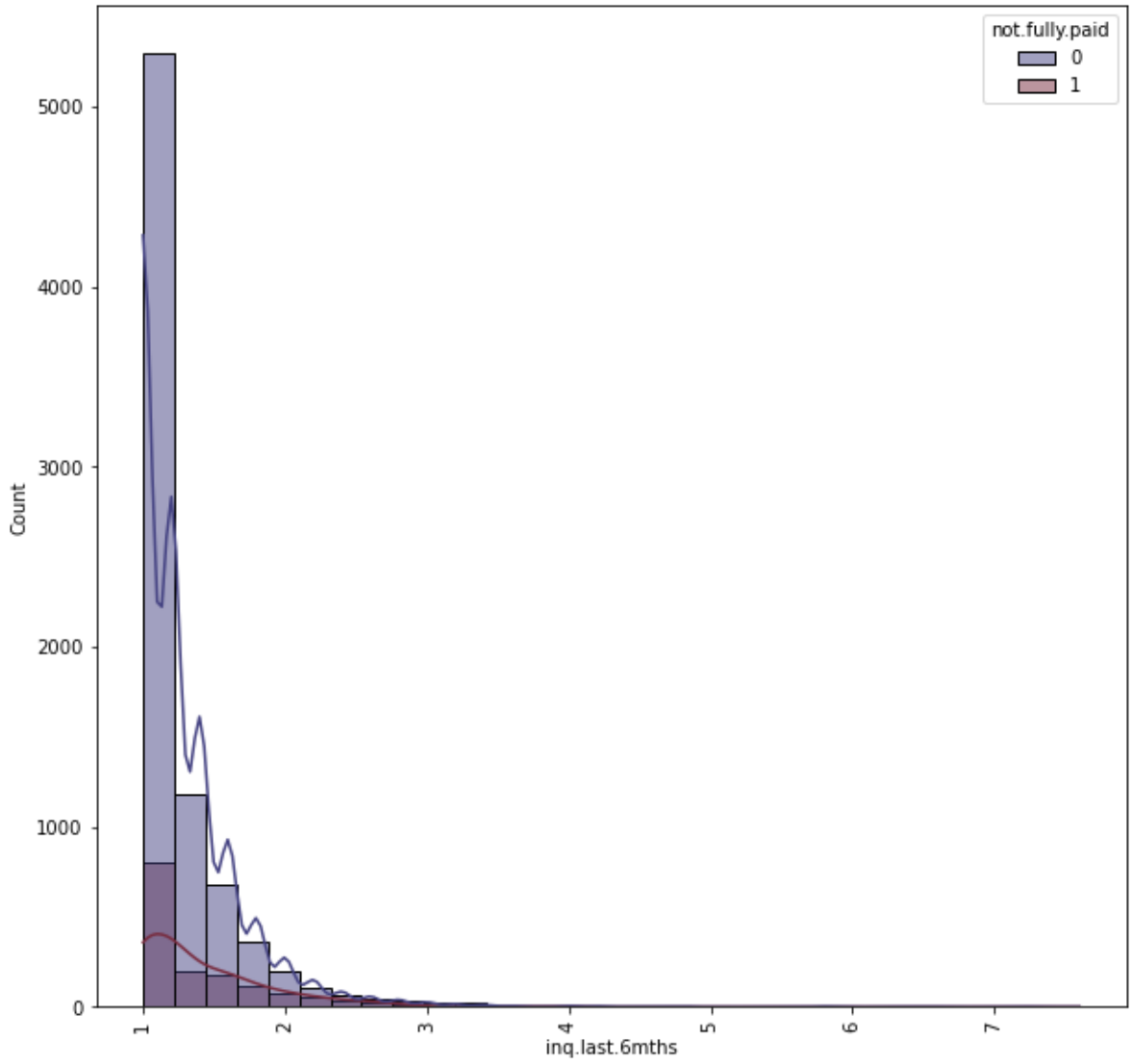
In [191]:

```
plt.figure(figsize=(10,10))
g = sns.histplot(x=df["inq.last.6mths"] , hue=df['not.fully.paid'] ,bins=30 , kde=True ,pal
g.set_xticklabels(labels=df2['inq.last.6mths'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\3958336566.py:3: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['inq.last.6mths'].value_counts().index,rotati
on=90)
```

Out[191]:

```
[Text(-5.0, 0, '0'),
 Text(0.0, 0, '1'),
 Text(5.0, 0, '2'),
 Text(10.0, 0, '3'),
 Text(15.0, 0, '4'),
 Text(20.0, 0, '5'),
 Text(25.0, 0, '6'),
 Text(30.0, 0, '7'),
 Text(35.0, 0, '8')]
```



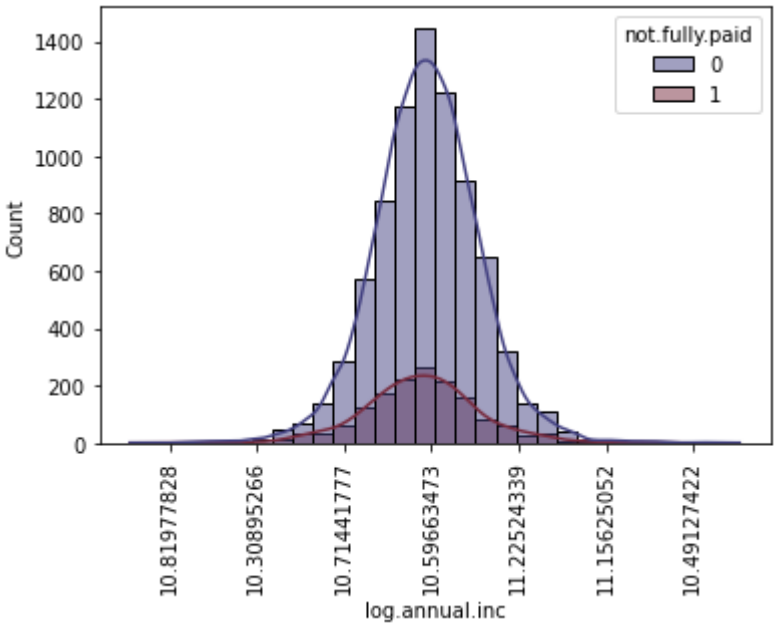
In [194]:

```
g = sns.histplot(x=df["log.annual.inc"] , hue=df['not.fully.paid'] ,bins=30,cbar=True , kde
g.set_xticklabels(labels=df2['log.annual.inc'].value_counts().index,rotation=90)
```

```
C:\Users\Ankita Sharma\AppData\Local\Temp\ipykernel_2256\3284744782.py:2: Us
erWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(labels=df2['log.annual.inc'].value_counts().index,rotati
on=90)
```

Out[194]:

```
[Text(7.0, 0, '11.00209984'),
 Text(8.0, 0, '10.81977828'),
 Text(9.0, 0, '10.30895266'),
 Text(10.0, 0, '10.71441777'),
 Text(11.0, 0, '10.59663473'),
 Text(12.0, 0, '11.22524339'),
 Text(13.0, 0, '11.15625052'),
 Text(14.0, 0, '10.49127422'),
 Text(15.0, 0, '10.77895629')]
```



In [263]:

```
#Standardising our data except our target variable and string variable.

from sklearn.preprocessing import StandardScaler
```

In [219]:

```
ss = StandardScaler()
num_data = df2.drop(['purpose' , 'not.fully.paid'] , axis=1)
df3 = pd.DataFrame(ss.fit_transform(num_data) , columns=num_data.columns)
df3
```

Out[219]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line
0	0.537902	-0.215472	2.389188	0.689278	0.980742	0.764447	0.453493
1	0.537902	-0.652936	-0.456180	0.261354	0.231084	-0.030731	-0.704204
2	0.537902	0.407358	0.200327	-0.869058	-0.153133	-0.693379	0.079665
3	0.537902	-0.886498	-0.768143	0.689278	-0.663016	0.101799	-0.728340
4	0.537902	0.663163	-1.049517	0.608445	0.329305	-1.090968	-0.179213
...	...	...	...	...	...	...	...
12040	0.537902	-0.652936	2.322609	-0.094595	-0.119911	0.499388	-1.132702
12041	0.537902	0.136723	1.519639	0.980111	1.086185	0.499388	1.888609
12042	-1.859075	0.399943	-0.572006	-0.594930	1.912397	-0.030731	0.966039
12043	0.537902	0.785505	-0.622911	-2.142806	-0.918679	-0.560850	-1.198644
12044	0.537902	-1.053327	0.286889	0.442132	-0.495462	1.162036	-0.619804

12045 rows × 19 columns

In [264]:

```
#preparing the labels for training our model , by one hot encoding our target variable df2[

from sklearn.preprocessing import OneHotEncoder
```

In [265]:

```
oe = OneHotEncoder()
labels = oe.fit_transform(np.array(df2['not.fully.paid']).reshape(-1,1))
labels=pd.DataFrame(labels.toarray(), columns=['fully.paid' , 'not.fully.paid'])
labels
```

Out[265]:

	fully.paid	not.fully.paid
0	1.0	0.0
1	1.0	0.0
2	1.0	0.0
3	1.0	0.0
4	1.0	0.0
...	...	...
12040	0.0	1.0
12041	0.0	1.0
12042	0.0	1.0
12043	0.0	1.0
12044	0.0	1.0

12045 rows × 2 columns

In [266]:

```
#splitting the dataset into training and testing sets.

from sklearn.model_selection import train_test_split
```

In [267]:

```
x=df3
y=labels
xtrain , xtest , ytrain , ytest = train_test_split(x , y , test_size=0.2 , random_state=42)
```

Preparing the Model

In [241]:

```
import tensorflow
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense , Dropout
```

In [248]:

```
model = Sequential()

model.add(Dense(units=64 , activation='relu'))

model.add(Dense(units=64 , activation='relu'))

model.add(Dense(units=64 , activation='relu'))

model.add(Dense(units=32 , activation='relu'))

model.add(Dense(units=2 , activation='sigmoid'))
```

In [249]:

```
model.compile(optimizer='adam' , loss=tensorflow.keras.losses.BinaryCrossentropy() , metric
```

In [250]:

```
model.fit(xtrain , ytrain , epochs=35 , batch_size=20 , validation_data=(xtest , ytest) )
```

```
Epoch 30/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1941 - bi
nary_accuracy: 0.9212 - val_loss: 0.6425 - val_binary_accuracy: 0.8103
Epoch 31/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1800 - bi
nary_accuracy: 0.9285 - val_loss: 0.6027 - val_binary_accuracy: 0.8257
Epoch 32/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1634 - bi
nary_accuracy: 0.9360 - val_loss: 0.6404 - val_binary_accuracy: 0.8138
Epoch 33/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1812 - bi
nary_accuracy: 0.9269 - val_loss: 0.6337 - val_binary_accuracy: 0.8198
Epoch 34/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1659 - bi
nary_accuracy: 0.9341 - val_loss: 0.6828 - val_binary_accuracy: 0.8084
Epoch 35/35
482/482 [==============================] - 1s 2ms/step - loss: 0.1715 - bi
nary_accuracy: 0.9350 - val_loss: 0.6189 - val_binary_accuracy: 0.8219
```

Out[250]:

In [ ]:

In [ ]: