1 数据集分析

1.1 数据中的特征数： 34

ID：每个贷款申请的唯一标识符。

year: 贷款申请提交的年份。

loan_limit: 借款人可以申请贷款的最大金额。

Gender: 借款人的性别。

approv_in_adv: 表示贷款是否事先获批。

loan_type: 贷款类型，如个人或商业。

loan_purpose: 贷款用途，如家庭装修或债务合并。

Credit_Worthiness: 衡量借款人偿还贷款能力的得分或评级。

open_credit: 借款人拥有的未还信用额度。

business_or_commercial: 表示贷款是用于商业还是个人用途。

loan_amount: 借款人申请的贷款金额。

rate_of_interest: 贷款利率的百分比。

Interest_rate_spread: 贷款利率与贷款时国债安全利率之间的差异。

Upfront_charges: 与贷款相关的任何预付费或费用。

term: 贷款还款期限。

Neg_ammortization: 表示贷款是否具有负面摊销功能。

interest_only：表明借款人在一段时间内只需支付贷款利息，然后再开始偿还贷款本金。

lump_sum_payment: 表示借款人可以进行一次性支付，以还清贷款本金。

property_value: 用作贷款抵押品的财产价值。

construction_type: 表示财产维修的类型。

occupancy_type: 表示财产是否为业主自住或非自住。

Secured_by: 表示用于担保贷款的抵押品类型。

total_units: 抵押物质产权益的个数。

income: 借款人的收入。

credit_type: 信用产品类型，如信用卡或汽车贷款。

Credit_Score: 借款人在申请贷款时的信用评分。

co-applicant_credit_type: 共同申请人的信用产品类型（如果有）。

age: 借款人申请贷款时的年龄。

submission_of_application: 贷款申请提交日期。

LTV: 贷款的贷款与价值比（贷款金额除以财产价值）。

Region: 财产所在地区。

Security_Type: 用于担保贷款的安全类型。

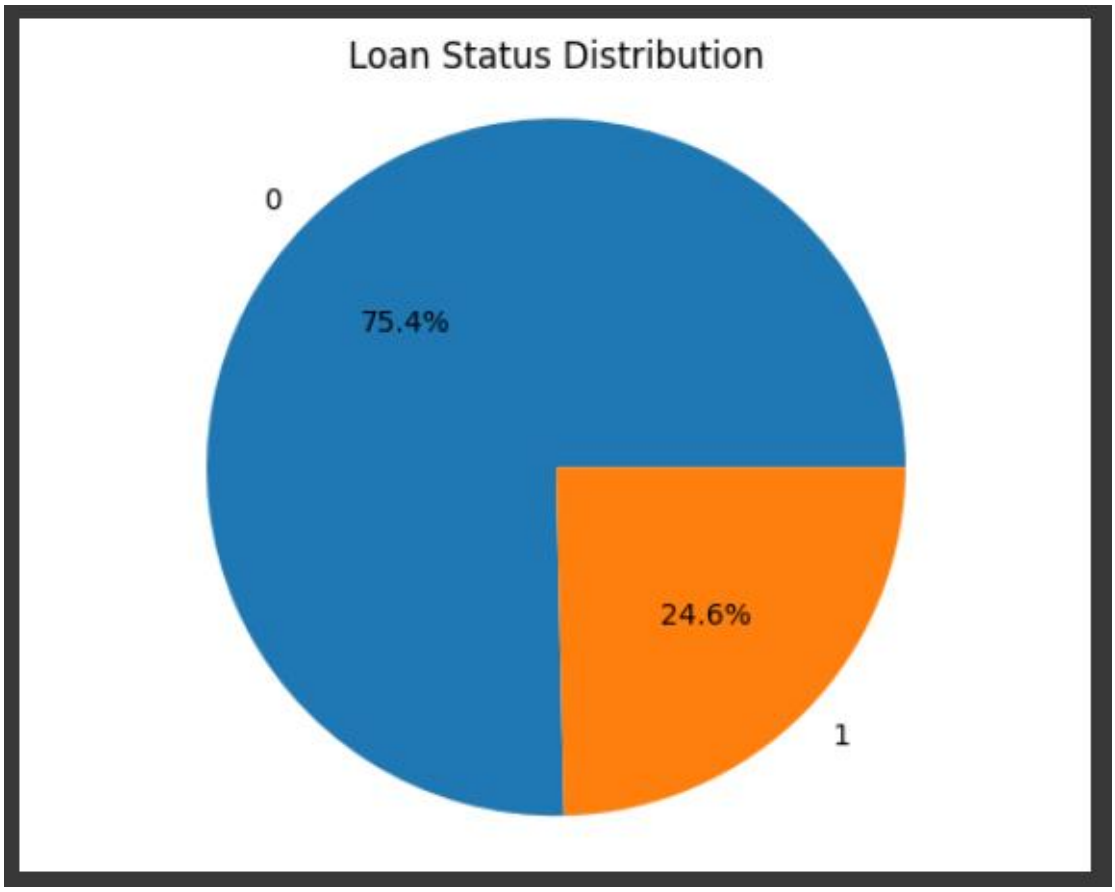
Status: 表示贷款申请的状态。

dtir1: 借款人的债务收入比（总月度债务支付除以月收入）。

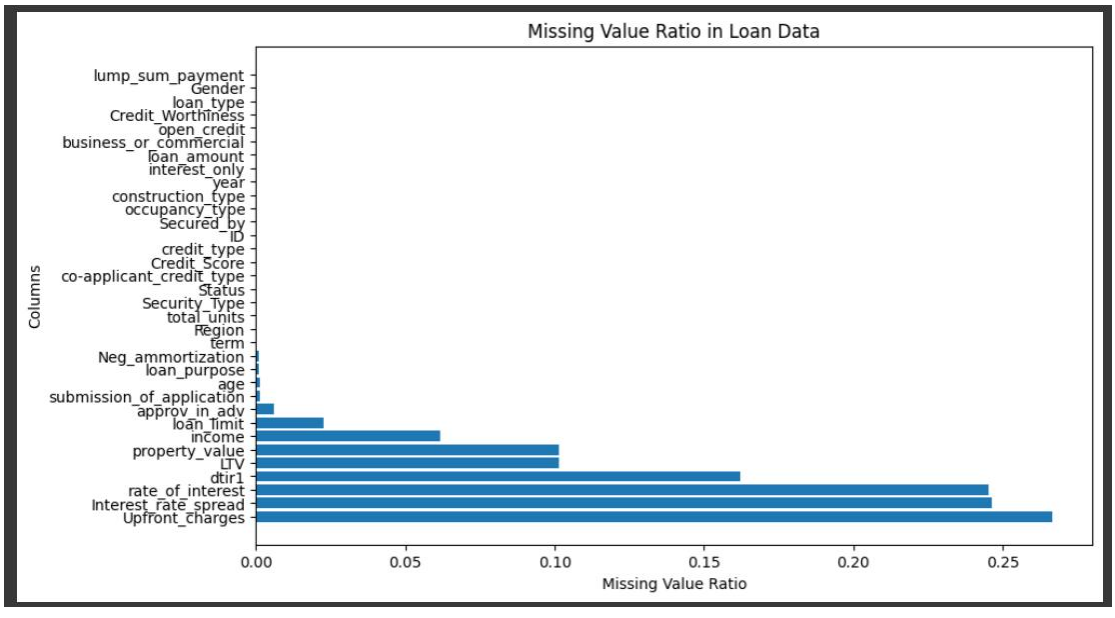
```
数据中的特征数： 34
数据中每个特征对应的数据类型：
 ID                          int64
year                         int64
loan_limit                  object
Gender                      object
approv_in_adv               object
loan_type                   object
loan_purpose                object
Credit_Worthiness           object
open_credit                 object
business_or_commercial      object
loan_amount                  int64
rate_of_interest           float64
Interest_rate_spread       float64
Upfront_charges            float64
term                       float64
Neg_ammortization           object
interest_only               object
lump_sum_payment            object
property_value             float64
construction_type           object
occupancy_type              object
Secured_by                  object
total_units                 object
income                     float64
credit_type                 object
Credit_Score                 int64
co-applicant_credit_type    object
age                         object
submission_of_application   object
LTV                        float64
Region                      object
Security_Type               object
Status                       int64
dtir1                      float64
dtype: object
```

1.2 贷款申请成功和失败的比例

0 表示贷款成功，1 代表贷款失败，可以看到成功和失败的比例大概为 3:1

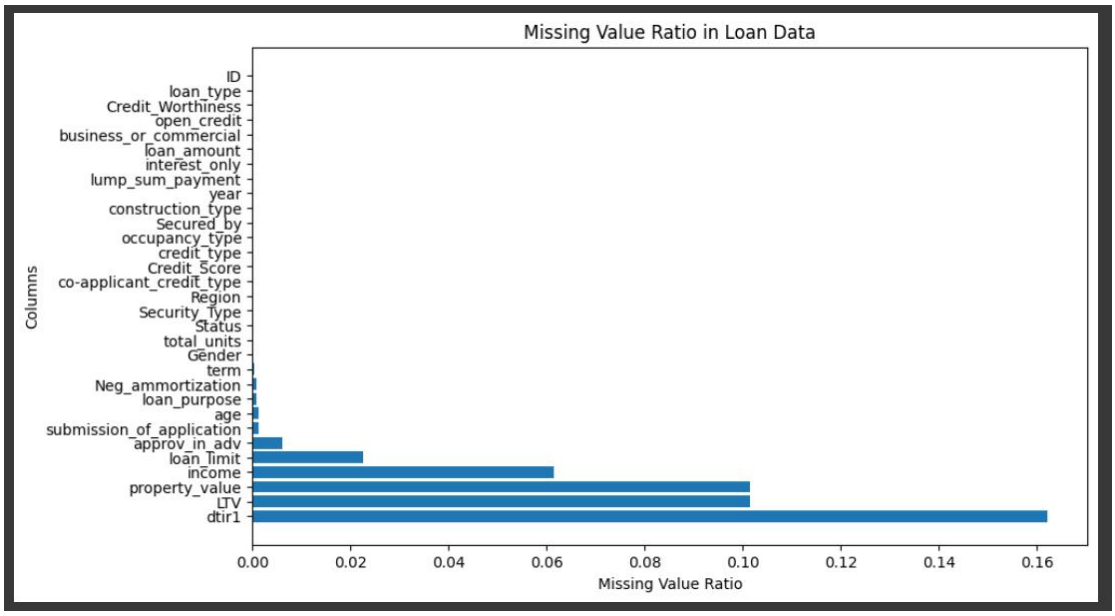


1.3 各特征的缺失值比例



对缺失值比例较高的几列进行分析：

Upfront_charges,Interest_rate_spread,rate_of_interest 这三列缺失的原因是因为这三个值是只有贷款成功的人才有的，而贷款失败的人是不会有的，因此在对放贷分析时，我们直接去除这三个列。去除三列后的结果。



dtir1 列，出现频率前十的值为如下：

```
37.0    6848
36.0    6553
44.0    6500
49.0    6309
43.0    5307
42.0    5121
41.0    4881
40.0    4699
39.0    4540
38.0    4461
Name: dtir1, dtype: int64
```

LTV 列，出现频率前十的值为如下：

```
81.250000    530
91.666667    499
80.038760    380
80.032468    328
94.956140    322
78.846154    317
78.645833    310
79.040404    309
80.063291    309
95.168067    306
Name: LTV, dtype: int64
```

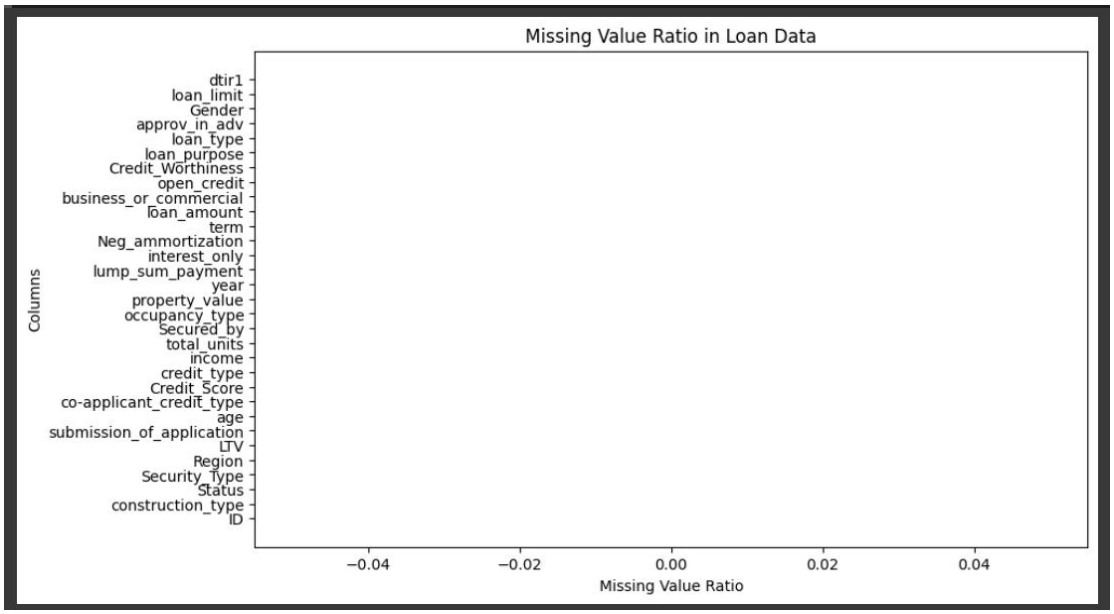
property_value 列，出现频率前十的值为如下：

```
308000.0    2792
258000.0    2763
358000.0    2679
408000.0    2537
328000.0    2524
278000.0    2513
268000.0    2497
228000.0    2493
238000.0    2408
288000.0    2398
Name: property_value, dtype: int64
```

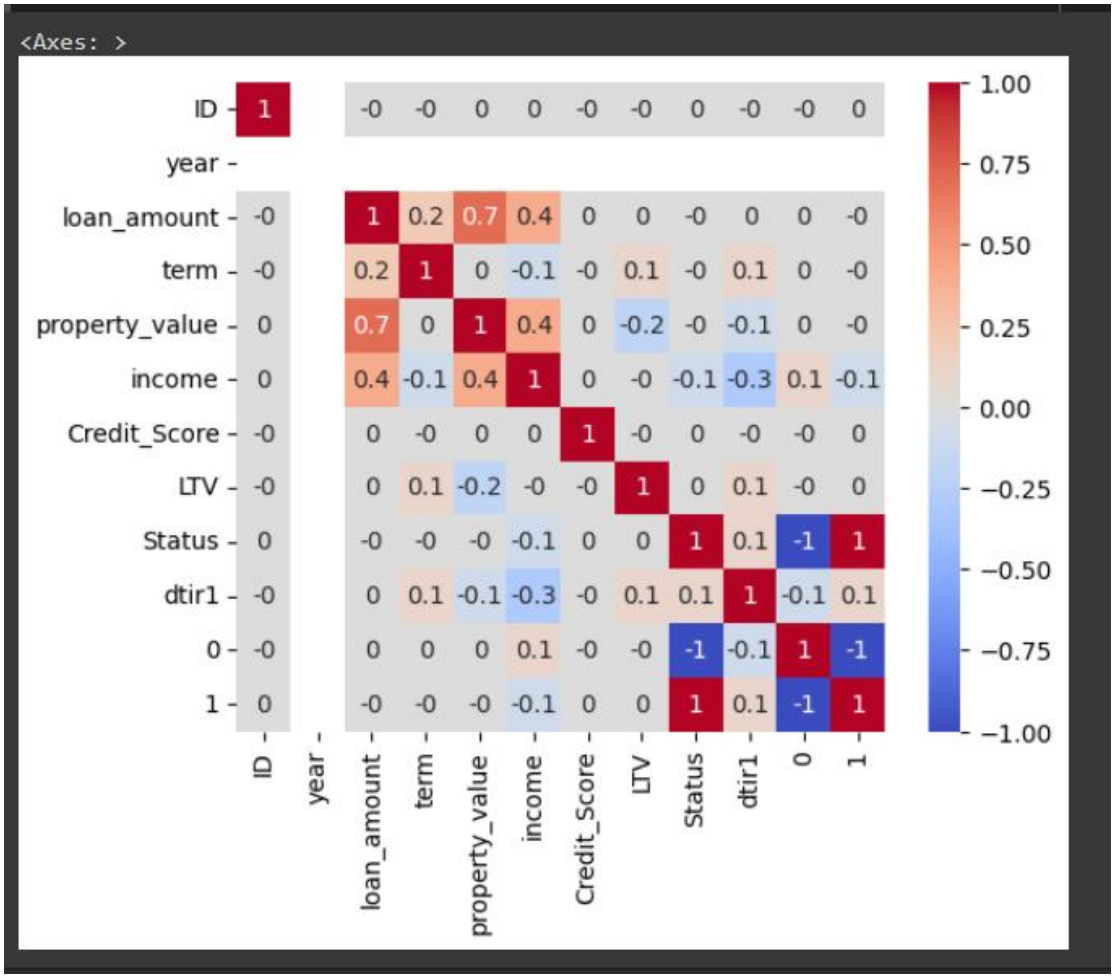
income 列，出现频率前十的值为如下：

```
0.0       1260
3600.0    1250
4200.0    1243
4800.0    1191
3120.0    1168
3720.0    1161
3900.0    1159
5400.0    1152
3300.0    1144
4500.0    1139
Name: income, dtype: int64
```

然后进行填充，对于 object 类型的列使用出现次数最多的值进行填充，# 对于数据类型的列使用平均值填充缺失值。填充后的结果如下：



对特征进行一下相关性分析：


<Axes: >
ID
year
loan_amount
term
property_value
income
Credit_Score
LTV
Status
dtir1
0
1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

结果分析：

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.8956	0.8946	0.6229	0.9303	0.7462	0.6878	0.7054	
gbc	Gradient Boosting Classifier	0.8880	0.8837	0.5853	0.9364	0.7203	0.6549	0.6832	
rf	Random Forest Classifier	0.8875	0.8850	0.5858	0.9330	0.7197	0.6539	0.6816	
ada	Ada Boost Classifier	0.8745	0.8678	0.5527	0.8996	0.6847	0.6118	0.6402	
et	Extra Trees Classifier	0.8688	0.8618	0.5187	0.9103	0.6608	0.5869	0.6226	
lda	Linear Discriminant Analysis	0.8625	0.8318	0.4538	0.9748	0.6192	0.5486	0.6084	
ridge	Ridge Classifier	0.8615	0.0000	0.4462	0.9822	0.6136	0.5433	0.6062	
knn	K Neighbors Classifier	0.8305	0.7365	0.4592	0.7577	0.5718	0.4740	0.4971	
dt	Decision Tree Classifier	0.8236	0.7716	0.6692	0.6348	0.6515	0.5335	0.5339	
qda	Quadratic Discriminant Analysis	0.8209	0.7486	0.5155	0.7500	0.5918	0.4823	0.5085	
lr	Logistic Regression	0.7536	0.5865	0.0000	0.0000	0.0000	0.0000	0.0000	
dummy	Dummy Classifier	0.7536	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	
nb	Naive Bayes	0.7506	0.6150	0.0305	0.4197	0.0563	0.0240	0.0534	
svm	SVM - Linear Kernel	0.6013	0.0000	0.3010	0.1603	0.1232	0.0013	0.0006	

LGBMClassifier

```
LGBMClassifier(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
               importance_type='split', learning_rate=0.1, max_depth=-1,
               min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
               n_estimators=100, n_jobs=-1, num_leaves=31, objective=None,
               random_state=1, reg_alpha=0.0, reg_lambda=0.0, silent='warn',
               subsample=1.0, subsample_for_bin=200000, subsample_freq=0)
```