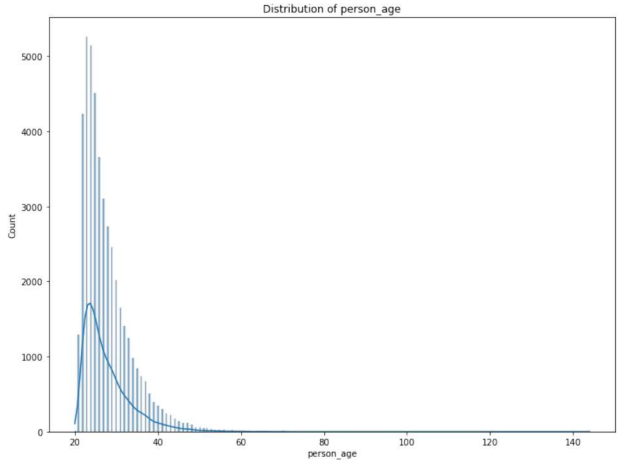
In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# 加載數據
df = pd.read_csv("C:/Users/user/Desktop/Cathay/data/loan_data.csv")

# 顯示前幾行和基本信息
print(df.head())
print(df.info())
```

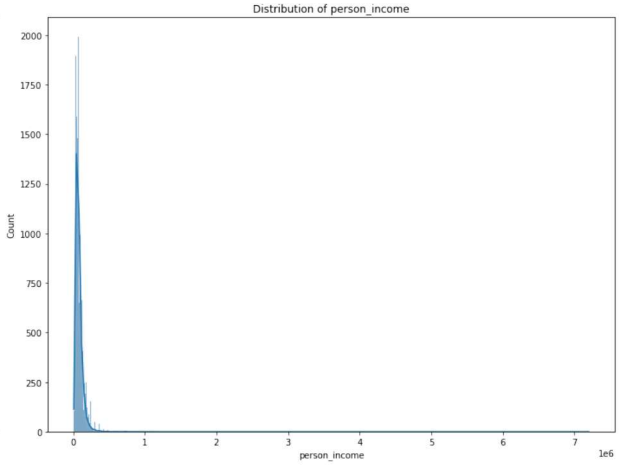
```
   person_age person_gender person_education  person_income  person_emp_exp  \
0        22.0        female           Master        71948.0               0
1        21.0        female      High School        12282.0               0
2        25.0        female      High School        12438.0               3
3        23.0        female         Bachelor        79753.0               0
4        24.0          male           Master        66135.0               1

  person_home_ownership  loan_amnt loan_intent  loan_int_rate  \
0                  RENT    35000.0    PERSONAL          16.02
1                   OWN     1000.0   EDUCATION          11.14
2              MORTGAGE     5500.0     MEDICAL          12.87
3                  RENT    35000.0     MEDICAL          15.23
4                  RENT    35000.0     MEDICAL          14.27

   loan_percent_income  cb_person_cred_hist_length  credit_score  \
0                 0.49                         3.0           561
1                 0.08                         2.0           504
2                 0.44                         3.0           635
3                 0.44                         2.0           675
4                 0.53                         4.0           586

  previous_loan_defaults_on_file  loan_status
0                             No            1
1                            Yes            0
2                             No            1
3                             No            1
4                             No            1
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45000 entries, 0 to 44999
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   person_age                      45000 non-null  float64
 1   person_gender                   45000 non-null  object
 2   person_education                45000 non-null  object
 3   person_income                   45000 non-null  float64
 4   person_emp_exp                  45000 non-null  int64
 5   person_home_ownership           45000 non-null  object
 6   loan_amnt                       45000 non-null  float64
 7   loan_intent                     45000 non-null  object
 8   loan_int_rate                   45000 non-null  float64
 9   loan_percent_income             45000 non-null  float64
 10  cb_person_cred_hist_length      45000 non-null  float64
 11  credit_score                    45000 non-null  int64
 12  previous_loan_defaults_on_file  45000 non-null  object
 13  loan_status                     45000 non-null  int64
dtypes: float64(6), int64(3), object(5)
memory usage: 4.8+ MB
None
```

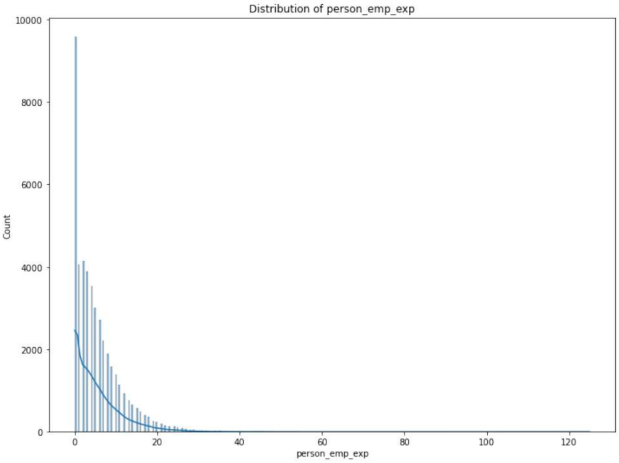
In [2]:

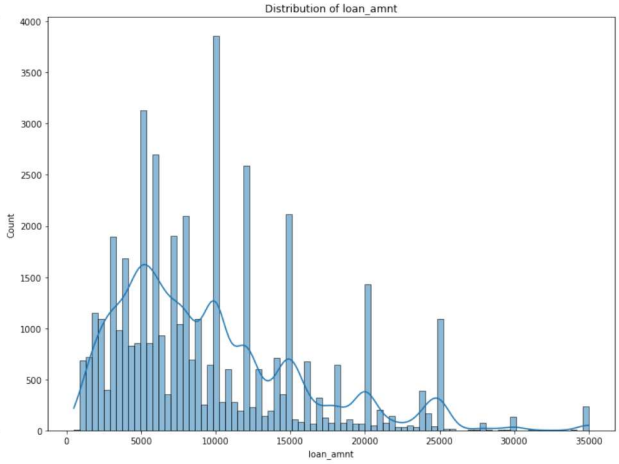
```python
import matplotlib.pyplot as plt
import seaborn as sns

# 數值特徵的分佈
numerical_features = ['person_age', 'person_income', 'person_emp_exp', 'loan_amnt',
                      'loan_int_rate', 'loan_percent_income', 'cb_person_cred_hist_ler
                      'credit_score']

fig, axes = plt.subplots(4, 2, figsize=(20, 30))
axes = axes.flatten()
```

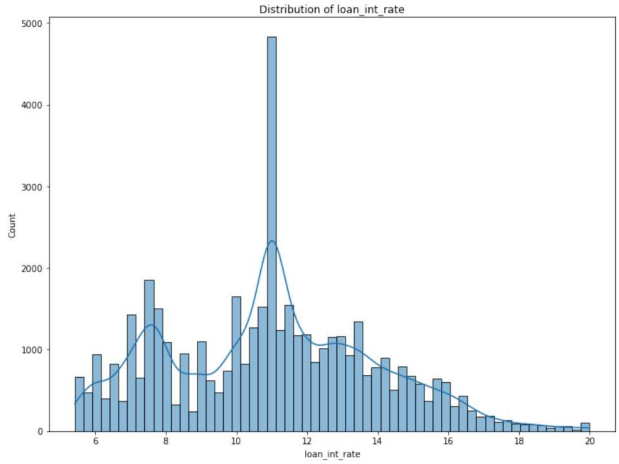
```
for i, feature in enumerate(numerical_features):
    sns.histplot(data=df, x=feature, kde=True, ax=axes[i])
    axes[i].set_title(f'Distribution of {feature}')

plt.tight_layout()
plt.show()

# 目標變量的分佈
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='loan_status')
plt.title('Distribution of Loan Status')
plt.show()

# 相關性矩陣
correlation_matrix = df[numerical_features + ['loan_status']].corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```

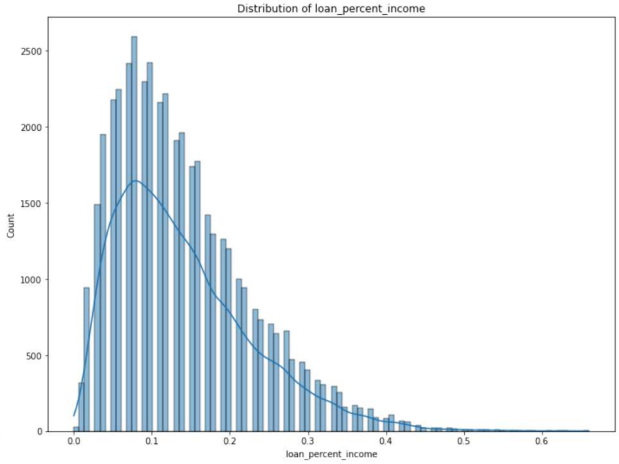

Distribution of person_age
Count
person_age

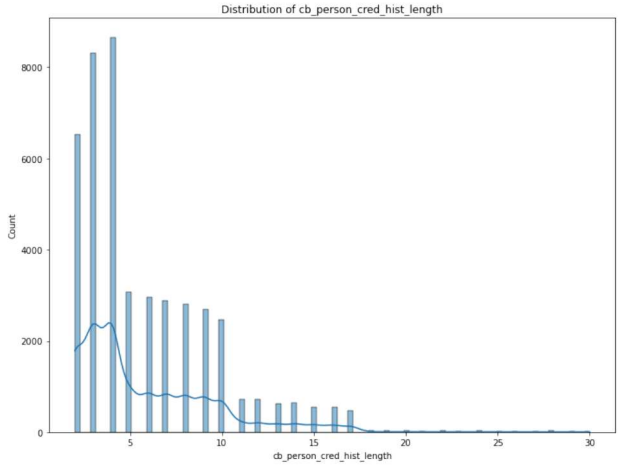

Distribution of person_income
Count
person_income
1e6

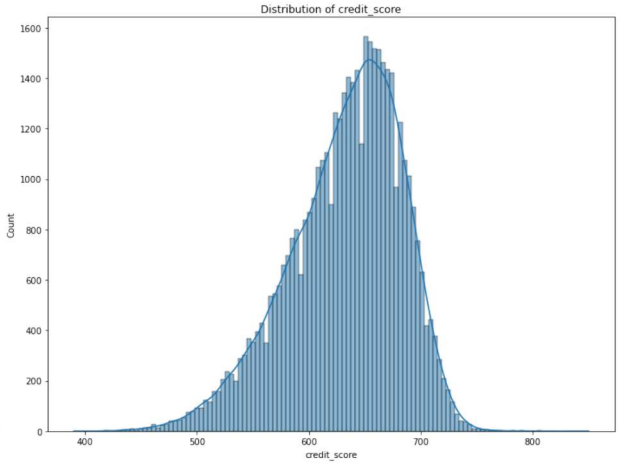

Distribution of person_emp_exp
Count
person_emp_exp

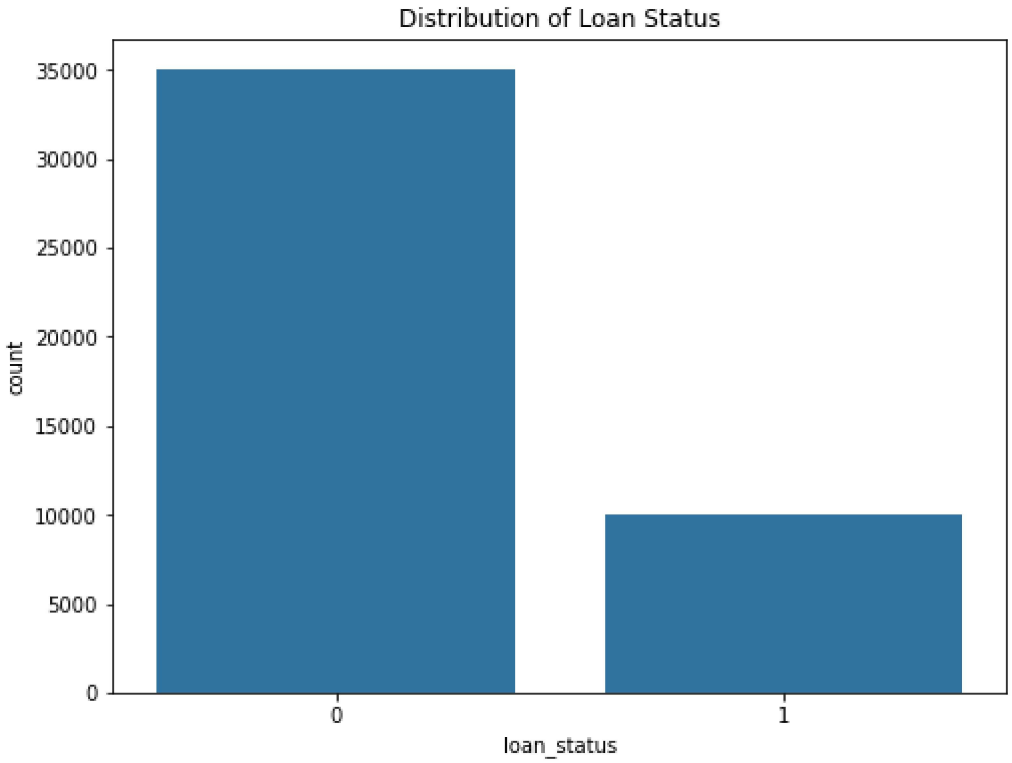

Distribution of loan_amnt
Count
loan_amnt

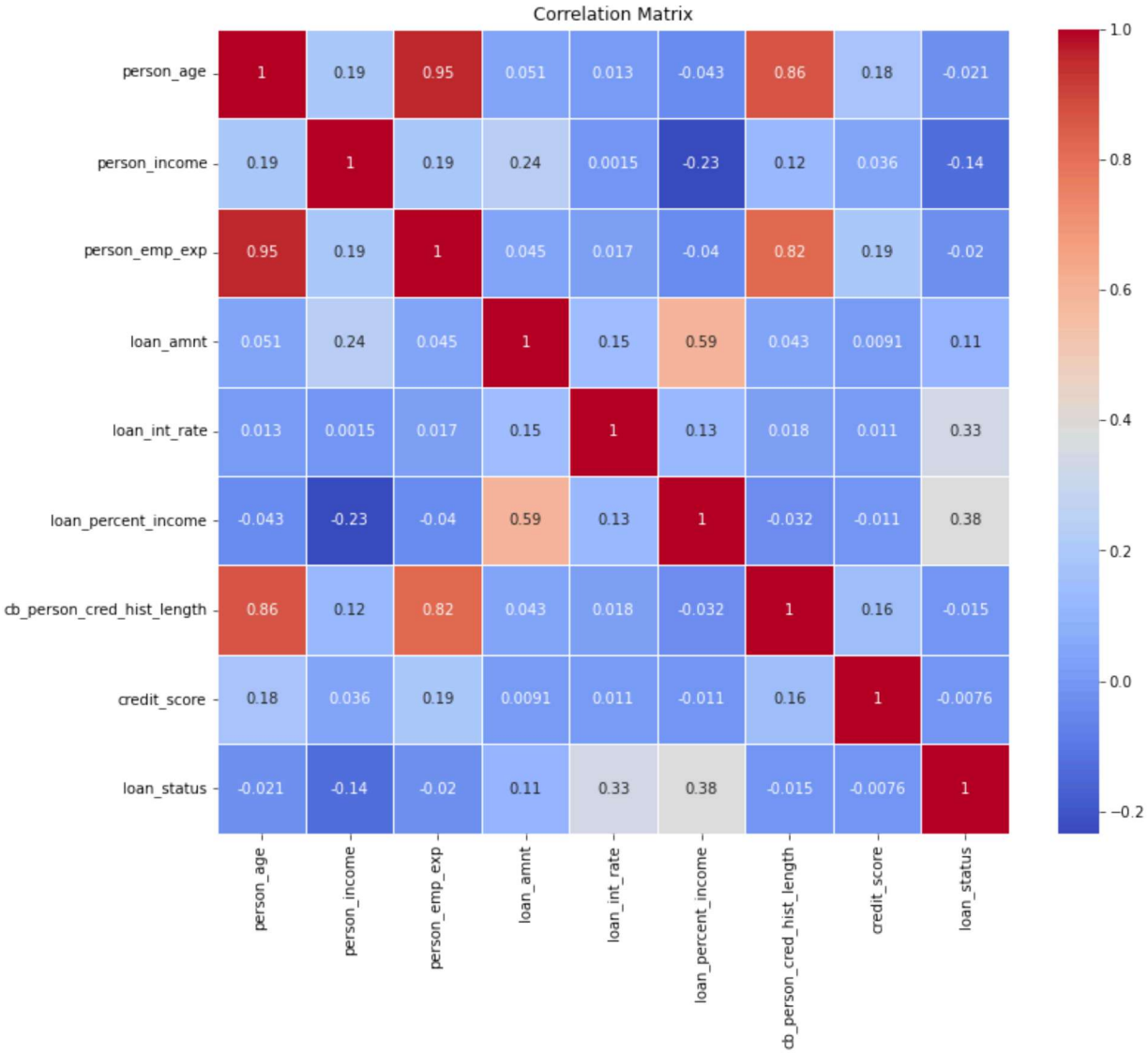

Distribution of loan_int_rate
Count
loan_int_rate


Distribution of loan_percent_income
Count
loan_percent_income

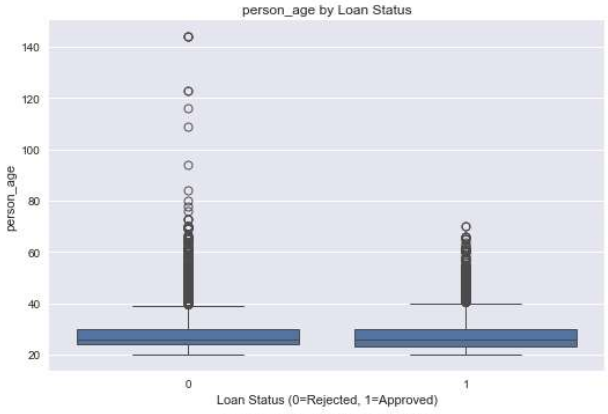

Distribution of cb_person_cred_hist_length
Count
cb_person_cred_hist_length

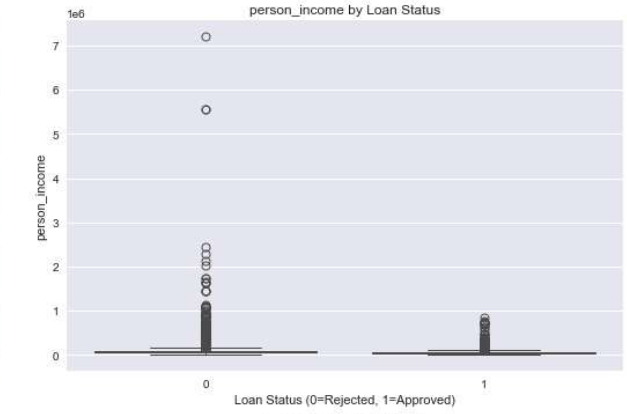

Distribution of credit_score
Count
credit_score

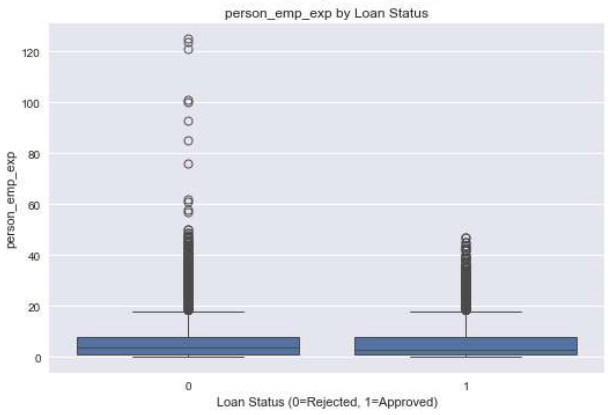

Distribution of Loan Status
count
35000
30000
25000
20000
15000
10000
5000
0
0
1
loan_status

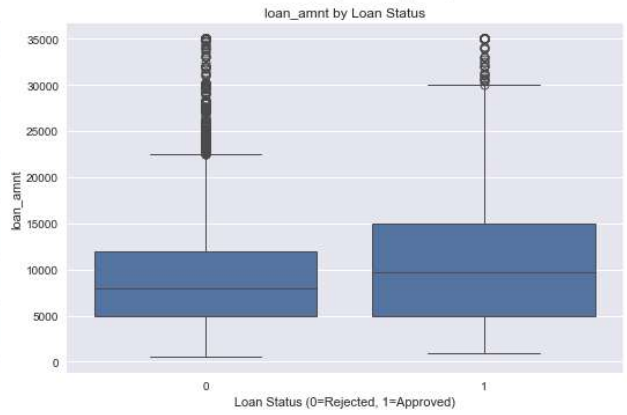

Correlation Matrix
person_age
person_income
person_emp_exp
loan_amnt
loan_int_rate
loan_percent_income
cb_person_cred_hist_length
credit_score
loan_status
1 0.19 0.95 0.051 0.013 -0.043 0.86 0.18 -0.021
0.19 1 0.19 0.24 0.0015 -0.23 0.12 0.036 -0.14
0.95 0.19 1 0.045 0.017 -0.04 0.82 0.19 -0.02
0.051 0.24 0.045 1 0.15 0.59 0.043 0.0091 0.11
0.013 0.0015 0.017 0.15 1 0.13 0.018 0.011 0.33
-0.043 -0.23 -0.04 0.59 0.13 1 -0.032 -0.011 0.38
0.86 0.12 0.82 0.043 0.018 -0.032 1 0.16 -0.015
0.18 0.036 0.19 0.0091 0.011 -0.011 0.16 1 -0.0076
-0.021 -0.14 -0.02 0.11 0.33 0.38 -0.015 -0.0076 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

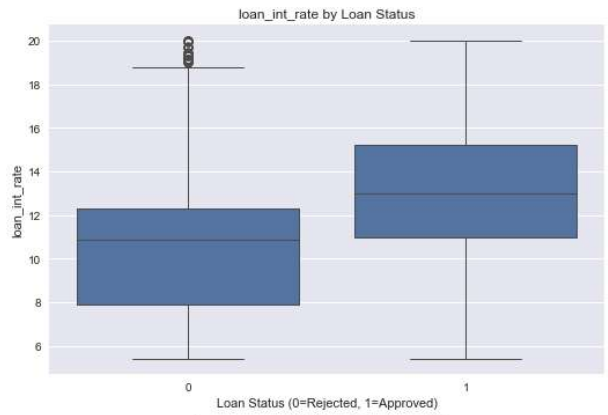
In [7]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# 設定圖表風格
plt.style.use('seaborn')

# 創建子圖
numerical_features = ['person_age', 'person_income', 'person_emp_exp', 'loan_amnt',
                      'loan_int_rate', 'loan_percent_income',
                      'cb_person_cred_hist_length', 'credit_score']

fig, axes = plt.subplots(4, 2, figsize=(15, 20))
axes = axes.ravel()

# 繪製每個數值特徵的箱型圖
for idx, col in enumerate(numerical_features):
    sns.boxplot(data=df, x='loan_status', y=col, ax=axes[idx])
    axes[idx].set_title(f'{col} by Loan Status')
    axes[idx].set_xlabel('Loan Status (0=Rejected, 1=Approved)')

plt.tight_layout()
plt.show()

# 計算每個特徵在不同貸款狀態下的統計摘要
summary_stats = df.groupby('loan_status')[numerical_features].describe()
print("\n數值特徵統計摘要:")
print(summary_stats)
```

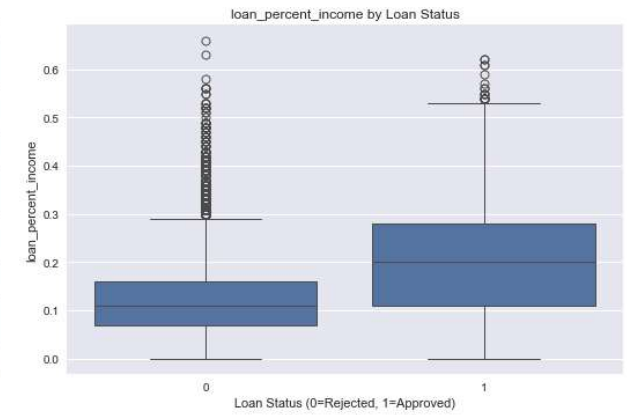

person_age by Loan Status
person_age
Loan Status (0=Rejected, 1=Approved)

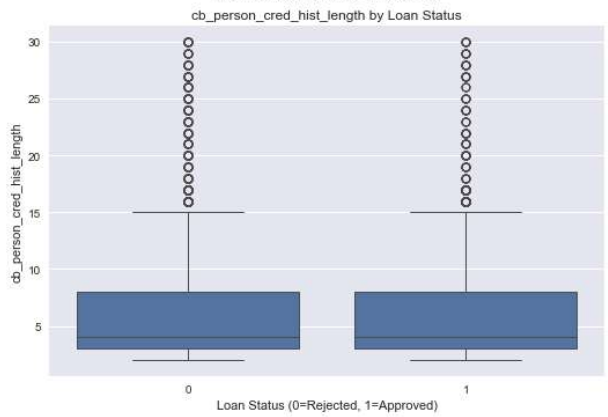

person_income by Loan Status
1e6
person_income
Loan Status (0=Rejected, 1=Approved)

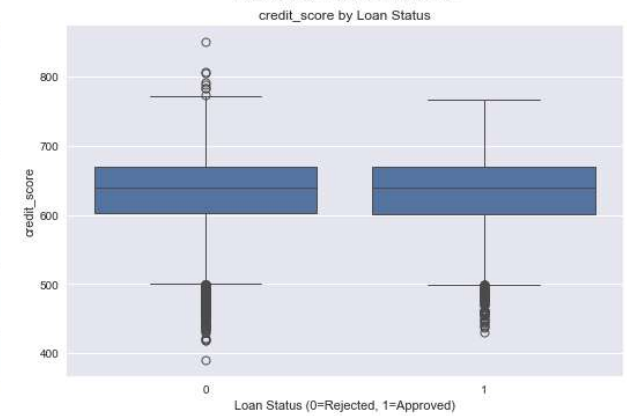

person_emp_exp by Loan Status
person_emp_exp
Loan Status (0=Rejected, 1=Approved)


loan_amnt by Loan Status
loan_amnt
Loan Status (0=Rejected, 1=Approved)


loan_int_rate by Loan Status
loan_int_rate
Loan Status (0=Rejected, 1=Approved)


loan_percent_income by Loan Status
loan_percent_income
Loan Status (0=Rejected, 1=Approved)


cb_person_cred_hist_length by Loan Status
cb_person_cred_hist_length
Loan Status (0=Rejected, 1=Approved)


credit_score by Loan Status
credit_score
Loan Status (0=Rejected, 1=Approved)

數值特徵統計摘要:

```
            person_age                                                    \
                 count       mean       std   min   25%   50%   75%    max
loan_status
0                35000.0  27.833571  6.073367  20.0  24.0  26.0  30.0  144.0
1                10000.0  27.521300  5.939063  20.0  23.0  26.0  30.0   70.0

            person_income                ... cb_person_cred_hist_length       \
                    count          mean  ...                        75%   max
loan_status                              ...
0                 35000.0  86157.040743  ...                        8.0  30.0
1                 10000.0  59886.096900  ...                        8.0  30.0

            credit_score                                                  \
                   count        mean        std    min    25%    50%    75%
loan_status
0                35000.0  632.814914  50.475294  390.0  602.0  640.0  670.0
1                10000.0  631.887200  50.293485  431.0  601.0  639.0  669.0


               max
loan_status
0            850.0
1            767.0

[2 rows x 64 columns]
```

In [4]:

```python
# 對類別特徵進行編碼
categorical_features = ['person_gender', 'person_education',
                        'person_home_ownership', 'loan_intent',
                        'previous_loan_defaults_on_file']

# 創建特徵工程的數據副本
df_encoded = df.copy()

# 對類別特徵進行獨熱編碼
df_encoded = pd.get_dummies(df_encoded, columns=categorical_features)

# 顯示處理後的特徵
print("特徵工程後的特徵列表:")
print(df_encoded.columns.tolist())
```

特徵工程後的特徵列表:

```
['person_age', 'person_income', 'person_emp_exp', 'loan_amnt', 'loan_int_rate', 'loan
_percent_income', 'cb_person_cred_hist_length', 'credit_score', 'loan_status', 'perso
n_gender_female', 'person_gender_male', 'person_education_Associate', 'person_educati
on_Bachelor', 'person_education_Doctorate', 'person_education_High School', 'person_e
ducation_Master', 'person_home_ownership_MORTGAGE', 'person_home_ownership_OTHER', 'p
erson_home_ownership_OWN', 'person_home_ownership_RENT', 'loan_intent_DEBTCONSOLIDATI
ON', 'loan_intent_EDUCATION', 'loan_intent_HOMEIMPROVEMENT', 'loan_intent_MEDICAL',
'loan_intent_PERSONAL', 'loan_intent_VENTURE', 'previous_loan_defaults_on_file_No',
'previous_loan_defaults_on_file_Yes']
```

In [5]:

```python
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.preprocessing import StandardScaler

# 準備特徵和目標變量
X = df_encoded.drop('loan_status', axis=1)
y = df_encoded['loan_status']
```

```
# 標準化數值特徵
numerical_features = ['person_age', 'person_income', 'person_emp_exp', 'loan_amnt',
                      'loan_int_rate', 'loan_percent_income',
                      'cb_person_cred_hist_length', 'credit_score']
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])

# 使用F檢驗進行特徵選擇
selector = SelectKBest(score_func=f_classif, k=15)
X_selected = selector.fit_transform(X, y)

# 獲取選擇的特徵名稱
selected_features_mask = selector.get_support()
selected_features = X.columns[selected_features_mask].tolist()

print("選擇的前15個最重要特徵:")
for feature, score in zip(X.columns[selected_features_mask], selector.scores_[selected
    print(f"{feature}: {score:.2f}")
```

```
選擇的前15個最重要特徵:
person_income: 845.53
loan_amnt: 528.21
loan_int_rate: 5574.45
loan_percent_income: 7824.79
person_home_ownership_MORTGAGE: 2148.03
person_home_ownership_OWN: 398.28
person_home_ownership_RENT: 3135.77
loan_intent_DEBTCONSOLIDATION: 320.76
loan_intent_EDUCATION: 185.10
loan_intent_HOMEIMPROVEMENT: 51.58
loan_intent_MEDICAL: 192.08
loan_intent_PERSONAL: 22.77
loan_intent_VENTURE: 335.22
previous_loan_defaults_on_file_No: 18824.73
previous_loan_defaults_on_file_Yes: 18824.73
```

In [6]:

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

# 使用選定的特徵
X = df_encoded[selected_features]
y = df_encoded['loan_status']

# 分割訓練集和測試集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

# 建立隨機森林模型
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# 預測
y_pred = rf_model.predict(X_test)

# 輸出模型評估報告
print(classification_report(y_test, y_pred))

# 輸出混淆矩陣
print("\n混淆矩陣:")
print(confusion_matrix(y_test, y_pred))
```

```
                   precision    recall  f1-score   support

               0       0.94      0.97      0.96      6990
               1       0.88      0.80      0.84      2010

        accuracy                           0.93      9000
       macro avg       0.91      0.88      0.90      9000
    weighted avg       0.93      0.93      0.93      9000


    混淆矩陣:
    [[6761  229]
     [ 400 1610]]
```

In [ ]: