Name : Gurjit Singh Sid: N01634963

Naive Bayes Classifier with Loan Dataset

1. Data Loading

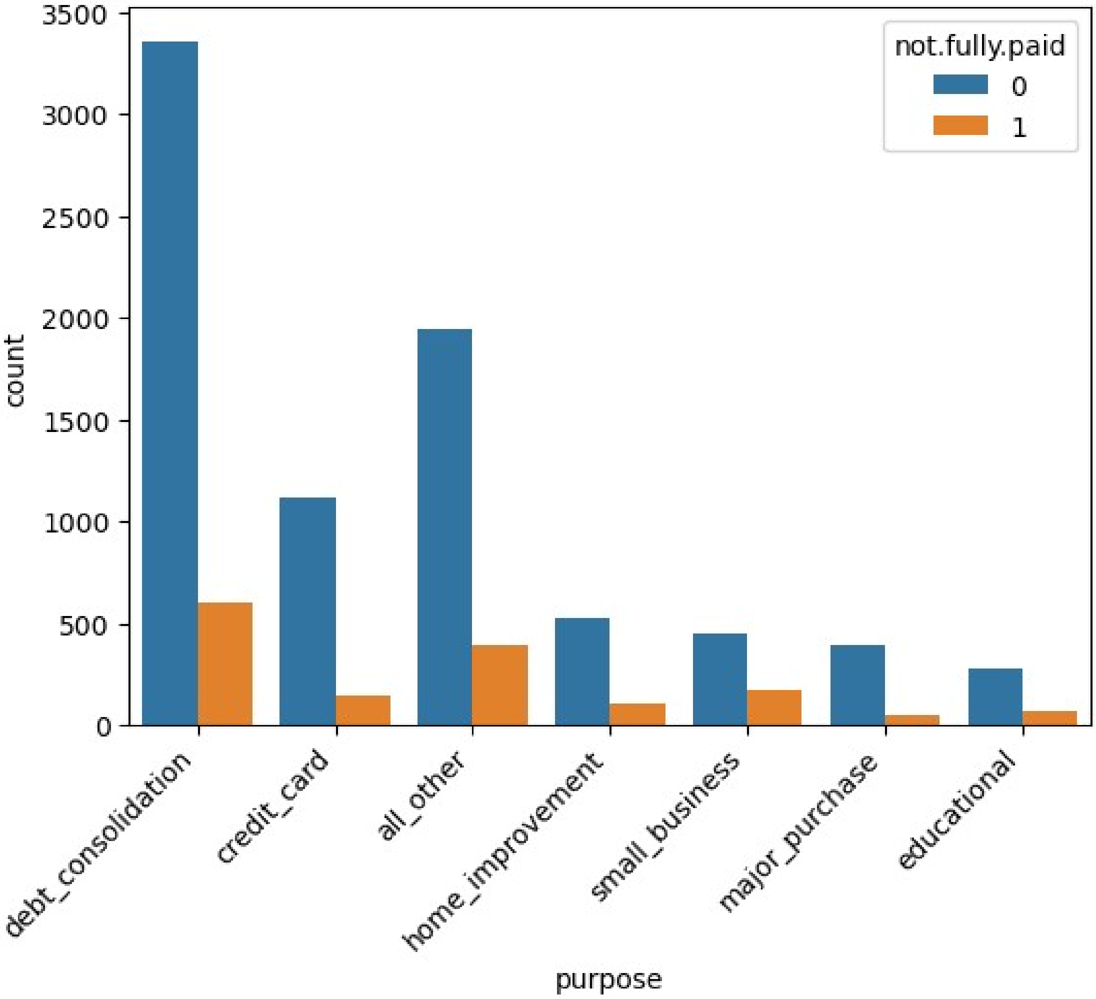
```
import pandas as pd


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

   credit.policy             purpose  int.rate  installment
log.annual.inc  \
0              1  debt_consolidation    0.1189       829.10
11.350407
1              1         credit_card    0.1071       228.22
11.082143
2              1  debt_consolidation    0.1357       366.86
10.373491
3              1  debt_consolidation    0.1008       162.34
11.350407
4              1         credit_card    0.1426       102.92
11.299732

     dti  fico  days.with.cr.line  revol.bal  revol.util
inq.last.6mths  \
0  19.48   737        5639.958333      28854        52.1
0
1  14.29   707        2760.000000      33623        76.7
0
2  11.63   682        4710.000000       3511        25.6
1
3   8.10   712        2699.958333      33667        73.2
1
4  14.97   667        4066.000000       4740        39.5
0

   delinq.2yrs  pub.rec  not.fully.paid
0            0        0               0
1            0        0               0
2            0        0               0
3            0        0               0
4            1        0               0
```

1. Data Exploration

```
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
```

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

# Convert the 'not.fully.paid' column to string (categorical)
df['not.fully.paid'] = df['not.fully.paid'].astype(str)

# Now plot
sns.countplot(data=df, x='purpose', hue='not.fully.paid')
plt.xticks(rotation=45, ha='right')
plt.show()
```



1. Data Processing

```
pre_df = pd.get_dummies(df,columns=['purpose'],drop_first=True)
pre_df.head()

   credit.policy  int.rate  installment  log.annual.inc    dti
fico  \
0              1    0.1189       829.10       11.350407  19.48   737

1              1    0.1071       228.22       11.082143  14.29   707

2              1    0.1357       366.86       10.373491  11.63   682

3              1    0.1008       162.34       11.350407   8.10   712

4              1    0.1426       102.92       11.299732  14.97   667


   days.with.cr.line  revol.bal  revol.util  inq.last.6mths
delinq.2yrs  \
0        5639.958333      28854        52.1                  0
0
1        2760.000000      33623        76.7                  0
0
2        4710.000000       3511        25.6                  1
0
3        2699.958333      33667        73.2                  1
0
4        4066.000000       4740        39.5                  0
1

   pub.rec not.fully.paid  purpose_credit_card
purpose_debt_consolidation  \
0        0              0                False
True
1        0              0                 True
False
2        0              0                False
True
3        0              0                False
True
4        0              0                 True
False

   purpose_educational  purpose_home_improvement
purpose_major_purchase  \
0                False                     False
False
1                False                     False
False
2                False                     False
False
3                False                     False
False
4                False                     False
False

   purpose_small_business
0                   False
1                   False
2                   False
3                   False
4                   False
```

After that, we will define feature (X) and target (y) variables, and split the dataset into training and testing sets.

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

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

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.33, random_state=125
)
```

1. Model Building and Training

```
from sklearn.naive_bayes import GaussianNB

model = GaussianNB()

model.fit(X_train, y_train);
```

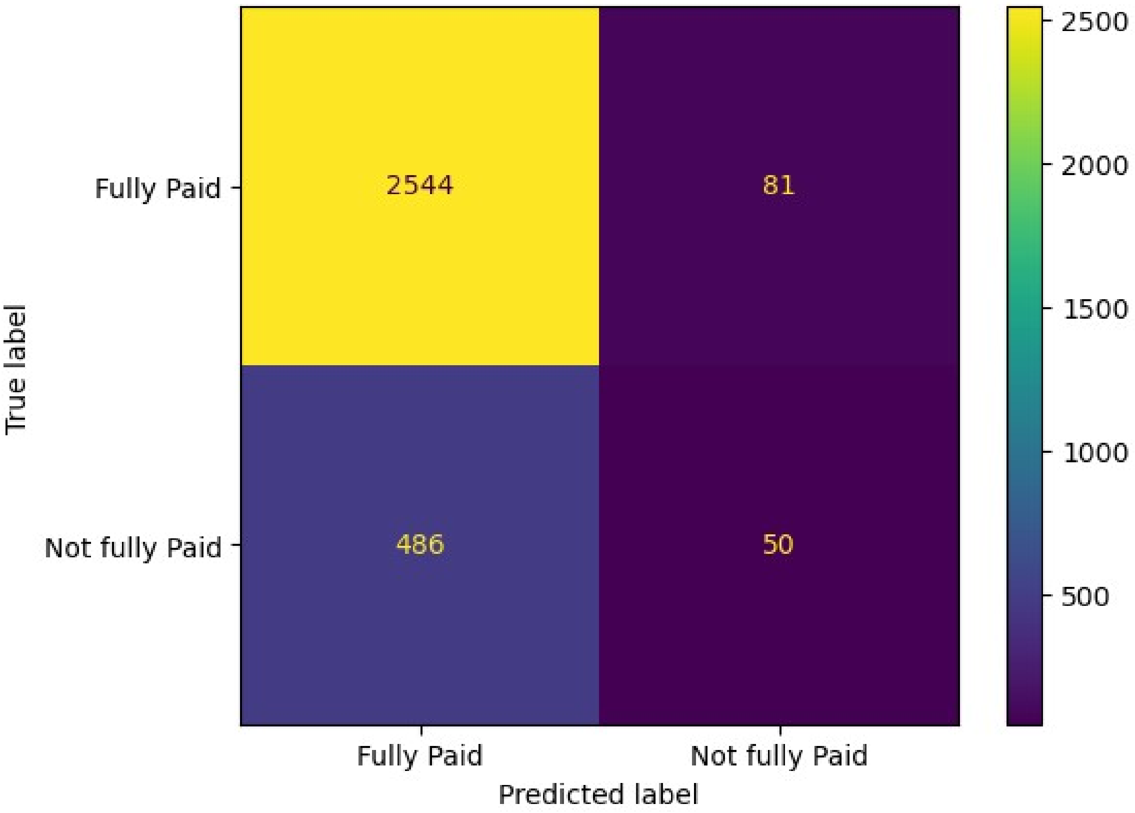
1. Model Evaluation

```
from sklearn.metrics import (
    accuracy_score,
    confusion_matrix,
    ConfusionMatrixDisplay,
    f1_score,
    classification_report,
)

y_pred = model.predict(X_test)

accuray = accuracy_score(y_pred, y_test)
f1 = f1_score(y_pred, y_test, average="weighted")

print("Accuracy:", accuray)
print("F1 Score:", f1)
```

```
Accuracy: 0.8206263840556786
F1 Score: 0.8686606980013266
```

1. Due to the imbalanced nature of the data, we can see that the confusion matrix tells a different story. On a minority target: `not fully paid`, we have more mislabeled.

```
labels = ["Fully Paid", "Not fully Paid"]
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=labels)
disp.plot();
```


Fully Paid
Not fully Paid
True label
2544
81
486
50
Fully Paid
Not fully Paid
Predicted label
2500
2000
1500
1000
500