In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
data=pd.read_csv(r"C:\Users\DELL\Downloads\ML Project - Naive Bayes Loan Status Classification U16955482770.c
```

In [3]:
```
data
```

Out[3]:

	ID	LIMIT_BAL	AGE	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2
0	1	2.0	24	3913.0	312.0	689.0	NaN	NaN	NaN	NaN	689.0
1	2	12.0	26	2682.0	1725.0	2682.0	3272.0	3455.0	3261.0	NaN	1.0
2	3	9.0	34	29239.0	1427.0	13559.0	14331.0	14948.0	15549.0	1518.0	15.0
3	4	5.0	37	4699.0	48233.0	49291.0	28314.0	28959.0	29547.0	2.0	219.0
4	5	5.0	57	8617.0	567.0	35835.0	294.0	19146.0	19131.0	2.0	36681.0
...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	22.0	39	188948.0	192815.0	28365.0	884.0	31237.0	1598.0	85.0	2.0
29996	29997	15.0	43	1683.0	1828.0	352.0	8979.0	519.0	NaN	1837.0	3526.0
29997	29998	3.0	37	3565.0	3356.0	2758.0	2878.0	2582.0	19357.0	NaN	NaN
29998	29999	8.0	41	-1645.0	78379.0	7634.0	52774.0	11855.0	48944.0	859.0	349.0
29999	3	5.0	46	47929.0	4895.0	49764.0	36535.0	32428.0	15313.0	278.0	18.0

30000 rows × 16 columns

In [4]:
```
data.isnull().sum()
```

Out[4]:
```
ID                  0
LIMIT_BAL           0
AGE                 0
BILL_AMT1        2008
BILL_AMT2        2506
BILL_AMT3        2870
BILL_AMT4        3195
BILL_AMT5        3506
BILL_AMT6        4020
PAY_AMT1         5249
PAY_AMT2         5396
PAY_AMT3         5968
PAY_AMT4         6408
PAY_AMT5         6703
PAY_AMT6         7173
Default Status      0
dtype: int64
```

In [5]:
```
data.columns
```

Out[5]:
```
Index(['ID', 'LIMIT_BAL', 'AGE', 'BILL_AMT1', 'BILL_AMT2', 'BILL_AMT3',
       'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1', 'PAY_AMT2',
       'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6', 'Default Status'],
      dtype='object')
```

In [6]:
```
lis=['ID', 'LIMIT_BAL', 'AGE', 'BILL_AMT1', 'BILL_AMT2', 'BILL_AMT3',
       'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1', 'PAY_AMT2',
       'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6', 'Default Status']
```

In [7]:
```
for i in lis:
    data[i]=data[i].fillna(data[i].mean())
```

...

In [8]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   ID              30000 non-null  int64  
 1   LIMIT_BAL       30000 non-null  float64
 2   AGE             30000 non-null  int64  
 3   BILL_AMT1       30000 non-null  float64
 4   BILL_AMT2       30000 non-null  float64
 5   BILL_AMT3       30000 non-null  float64
 6   BILL_AMT4       30000 non-null  float64
 7   BILL_AMT5       30000 non-null  float64
 8   BILL_AMT6       30000 non-null  float64
 9   PAY_AMT1        30000 non-null  float64
 10  PAY_AMT2        30000 non-null  float64
 11  PAY_AMT3        30000 non-null  float64
 12  PAY_AMT4        30000 non-null  float64
 13  PAY_AMT5        30000 non-null  float64
 14  PAY_AMT6        30000 non-null  float64
 15  Default Status  30000 non-null  object 
dtypes: float64(13), int64(2), object(1)
memory usage: 3.7+ MB
```

In [9]:
```python
data.describe()
```

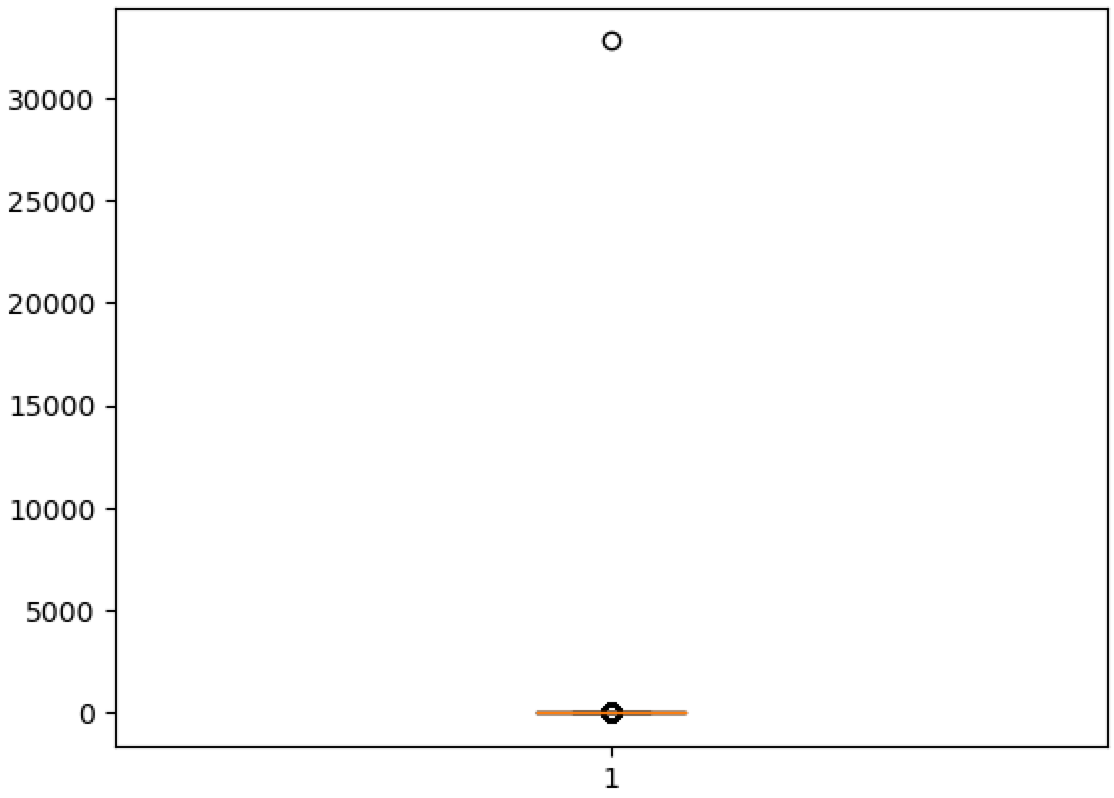
Out[9]:

	ID	LIMIT_BAL	AGE	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILI
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.00000	30000.000000	30000
mean	10666.660700	14.648867	32.427900	33989.494570	33938.750418	32617.867011	30709.50416	29015.123424	28526
std	9698.091793	189.496507	12.718991	59472.881131	58305.224138	55777.906879	51825.29715	48612.594594	48042
min	1.000000	1.000000	3.000000	-154973.000000	-69777.000000	-157264.000000	-81334.00000	-81334.000000	-94625
25%	1850.500000	4.000000	26.000000	1788.000000	1847.750000	1862.000000	1782.00000	1718.000000	1724
50%	7483.500000	9.000000	33.000000	11569.000000	12637.000000	13255.000000	13350.00000	12847.000000	13398
75%	18746.250000	21.000000	41.000000	33989.494570	33938.750418	32617.867011	30709.50416	29015.123424	28526
max	29999.000000	32768.000000	79.000000	964511.000000	983931.000000	693131.000000	891586.00000	927171.000000	961664

In [10]:
```python
import matplotlib.pyplot as plt
```

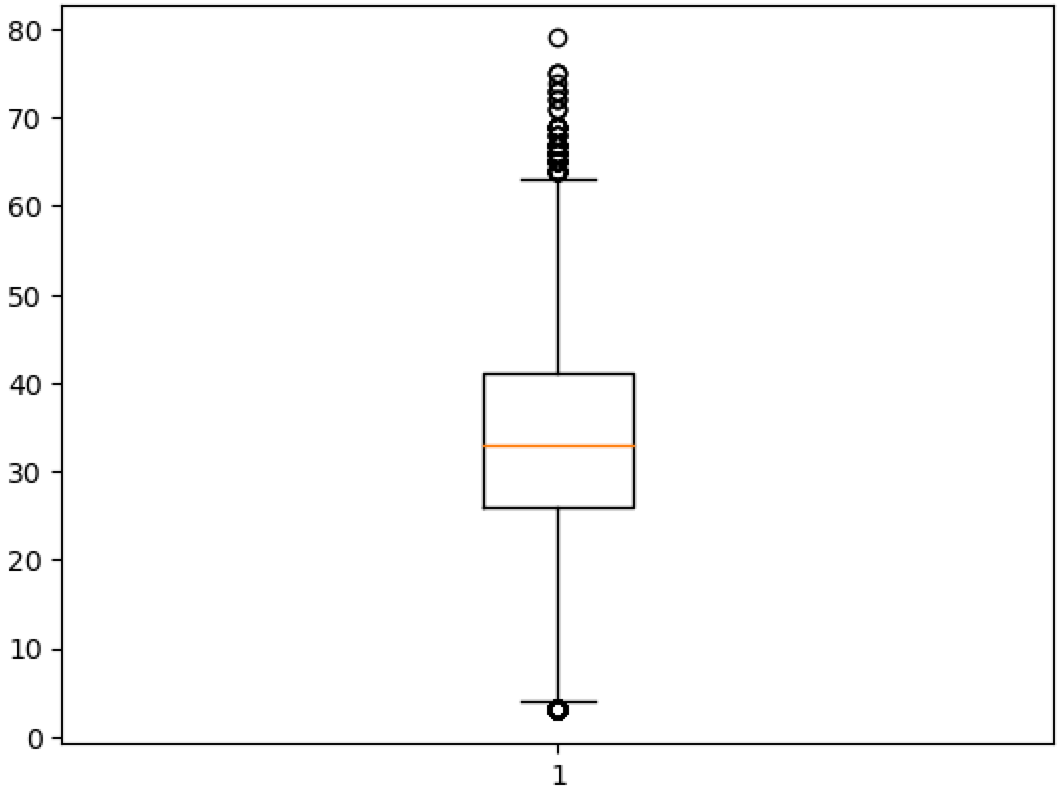
In [11]:
```
plt.boxplot(data["LIMIT_BAL"])
```

Out[11]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x196e5d9dd90>,
  <matplotlib.lines.Line2D at 0x196e5d9ea90>],
 'caps': [<matplotlib.lines.Line2D at 0x196e5d6c250>,
  <matplotlib.lines.Line2D at 0x196e5dac2d0>],
 'boxes': [<matplotlib.lines.Line2D at 0x196e5d668d0>],
 'medians': [<matplotlib.lines.Line2D at 0x196e5dace10>],
 'fliers': [<matplotlib.lines.Line2D at 0x196e5dad890>],
 'means': []}
```



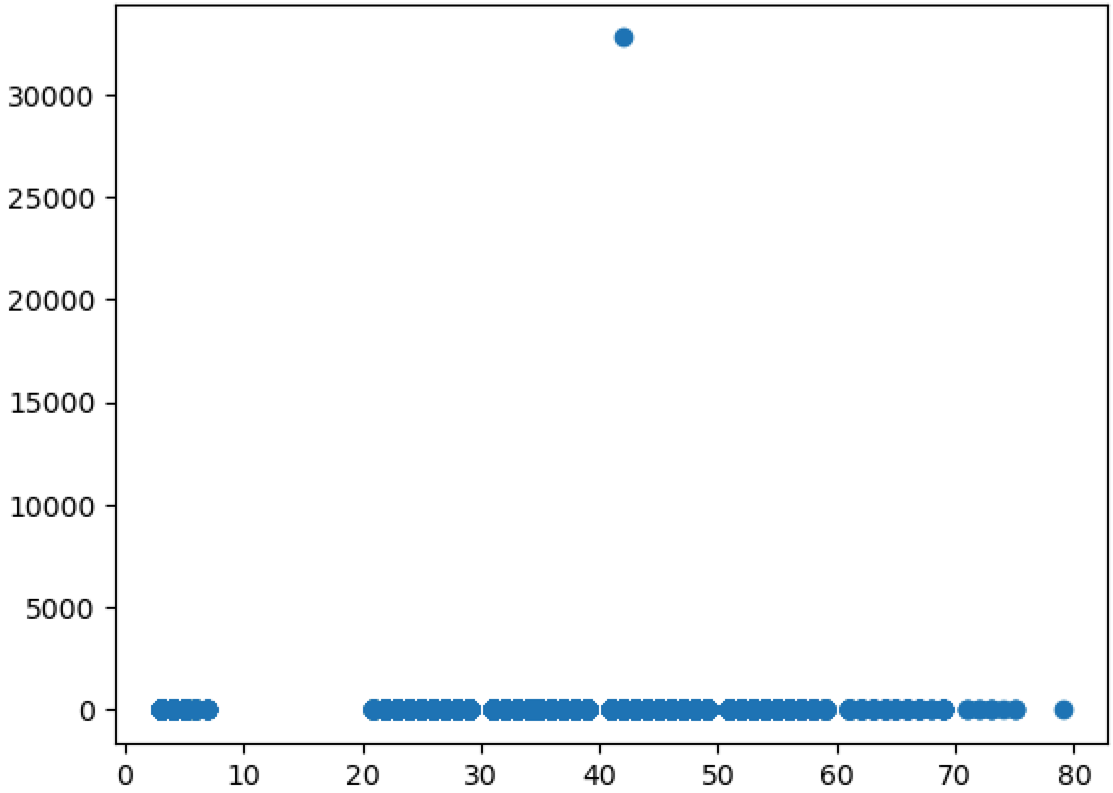
In [12]:
```
plt.boxplot(data["AGE"])
```

Out[12]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x196e664dbd0>,
  <matplotlib.lines.Line2D at 0x196e664e890>],
 'caps': [<matplotlib.lines.Line2D at 0x196e5daf490>,
  <matplotlib.lines.Line2D at 0x196e664ff90>],
 'boxes': [<matplotlib.lines.Line2D at 0x196e5dbe8d0>],
 'medians': [<matplotlib.lines.Line2D at 0x196e665cad0>],
 'fliers': [<matplotlib.lines.Line2D at 0x196e665d590>],
 'means': []}
```



In [13]:
```
plt.scatter(data['AGE'],data['LIMIT_BAL'])
```

Out[13]:
```
<matplotlib.collections.PathCollection at 0x196df341390>
```

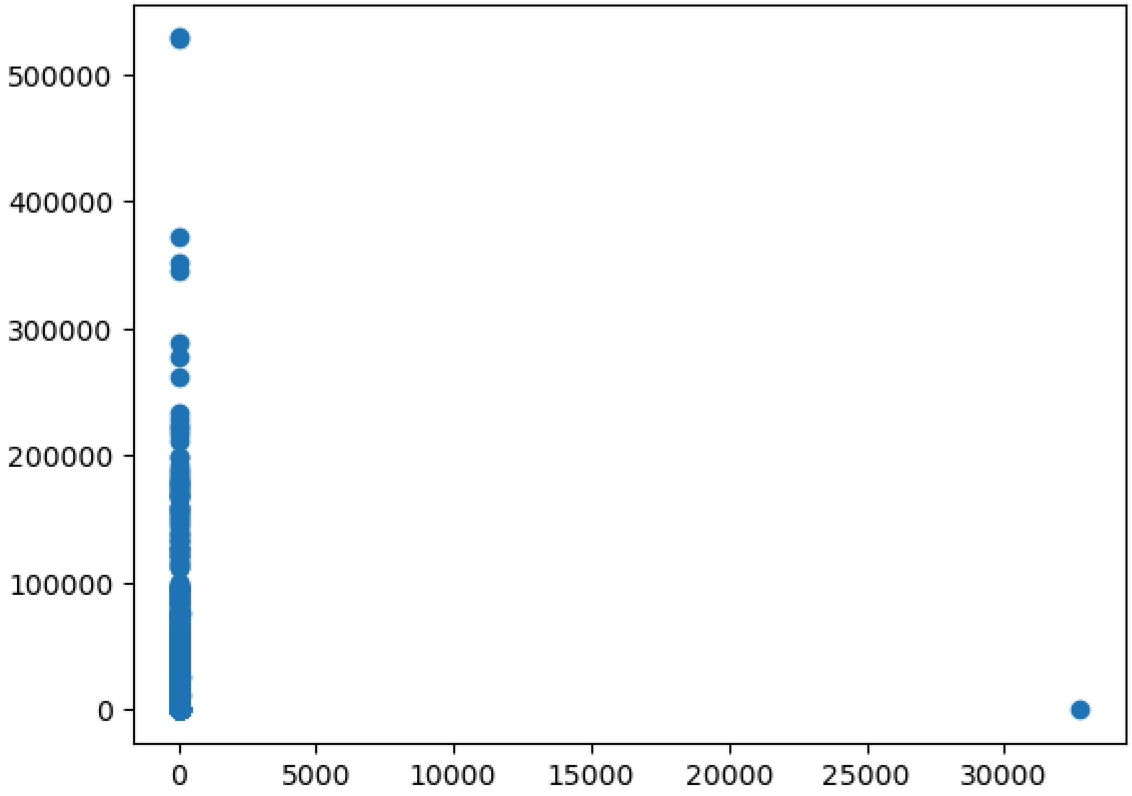


In [14]:

```
plt.scatter(data['LIMIT_BAL'],data['PAY_AMT6'])
```

Out[14]:

```
<matplotlib.collections.PathCollection at 0x196e5e6e650>
```



In [15]:

```
data.shape
```

Out[15]:

```
(30000, 16)
```

In [16]:

```
from sklearn.preprocessing import LabelEncoder
```

In [17]:

```
enc=LabelEncoder()
```

In [18]:
```
data["Default Status"]=enc.fit_transform(data["Default Status"])
```

In [19]:
```
data.head()
```

Out[19]:

	ID	LIMIT_BAL	AGE	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PA
0	1	2.0	24	3913.0	312.0	689.0	30709.50416	29015.123424	28526.276559	2613.957537	689.0	2584
1	2	12.0	26	2682.0	1725.0	2682.0	3272.00000	3455.000000	3261.000000	2613.957537	1.0	1
2	3	9.0	34	29239.0	1427.0	13559.0	14331.00000	14948.000000	15549.000000	1518.000000	15.0	1
3	4	5.0	37	4699.0	48233.0	49291.0	28314.00000	28959.000000	29547.000000	2.000000	219.0	12
4	5	5.0	57	8617.0	567.0	35835.0	294.00000	19146.000000	19131.000000	2.000000	36681.0	1

In [20]:
```
x=data.drop("Default Status",axis=1)
y=data["Default Status"]
```

In [21]:
```
from sklearn.model_selection import train_test_split
```

In [22]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.8)
```

In [25]:
```
from sklearn.naive_bayes import GaussianNB
```

In [26]:
```
clf=GaussianNB()
clf.fit(x_train,y_train)
```

Out[26]:
```
GaussianNB()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [27]:
```
clf.score(x_train,y_train)
```

Out[27]:
```
0.7541666666666667
```

In [28]:
```
clf.score(x_test,y_test)
```

Out[28]:
```
0.7459583333333333
```

In [30]:
```
y_pred=clf.predict(x_test)
```

In [35]:
```
from sklearn.metrics import classification_report
```

In [36]:
```
print(classification_report(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.16      0.04      0.06      5325
           1       0.78      0.95      0.85     18675

    accuracy                           0.75     24000
   macro avg       0.47      0.49      0.46     24000
weighted avg       0.64      0.75      0.68     24000

              precision    recall  f1-score   support

           0       0.16      0.04      0.06      5325
           1       0.78      0.95      0.85     18675

    accuracy                           0.75     24000
   macro avg       0.47      0.49      0.46     24000
weighted avg       0.64      0.75      0.68     24000
```

In [ ]: