In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
df = pd.read_csv('application_record.csv')
df = pd.DataFrame(df)
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 438557 entries, 0 to 438556
Data columns (total 19 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   ID                   438557 non-null  int64  
 1   CODE_GENDER          438557 non-null  object 
 2   FLAG_OWN_CAR         438557 non-null  object 
 3   FLAG_OWN_REALTY      438557 non-null  object 
 4   CNT_CHILDREN         438557 non-null  int64  
 5   AMT_INCOME_TOTAL     438557 non-null  float64
 6   NAME_INCOME_TYPE     438557 non-null  object 
 7   NAME_EDUCATION_TYPE  438557 non-null  object 
 8   NAME_FAMILY_STATUS   438557 non-null  object 
 9   NAME_HOUSING_TYPE    438557 non-null  object 
 10  DAYS_BIRTH           438557 non-null  int64  
 11  DAYS_EMPLOYED        438557 non-null  int64  
 12  FLAG_MOBIL           438557 non-null  int64  
 13  FLAG_WORK_PHONE      438557 non-null  int64  
 14  FLAG_PHONE           438557 non-null  int64  
 15  FLAG_EMAIL           438557 non-null  int64  
 16  OCCUPATION_TYPE      304354 non-null  object 
 17  CNT_FAM_MEMBERS      438557 non-null  int64  
 18  STATUS               36457 non-null   object 
dtypes: float64(1), int64(9), object(9)
memory usage: 63.6+ MB
```

In [82]:
```
df.shape
```

Out[82]:
```
(438557, 19)
```

In [8]:
```
df[df.duplicated()]
```

Out[8]:

| ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AMT_INCOME_TO |
|---|---|---|---|---|---|

In [9]:
```
df.isnull().sum()
```

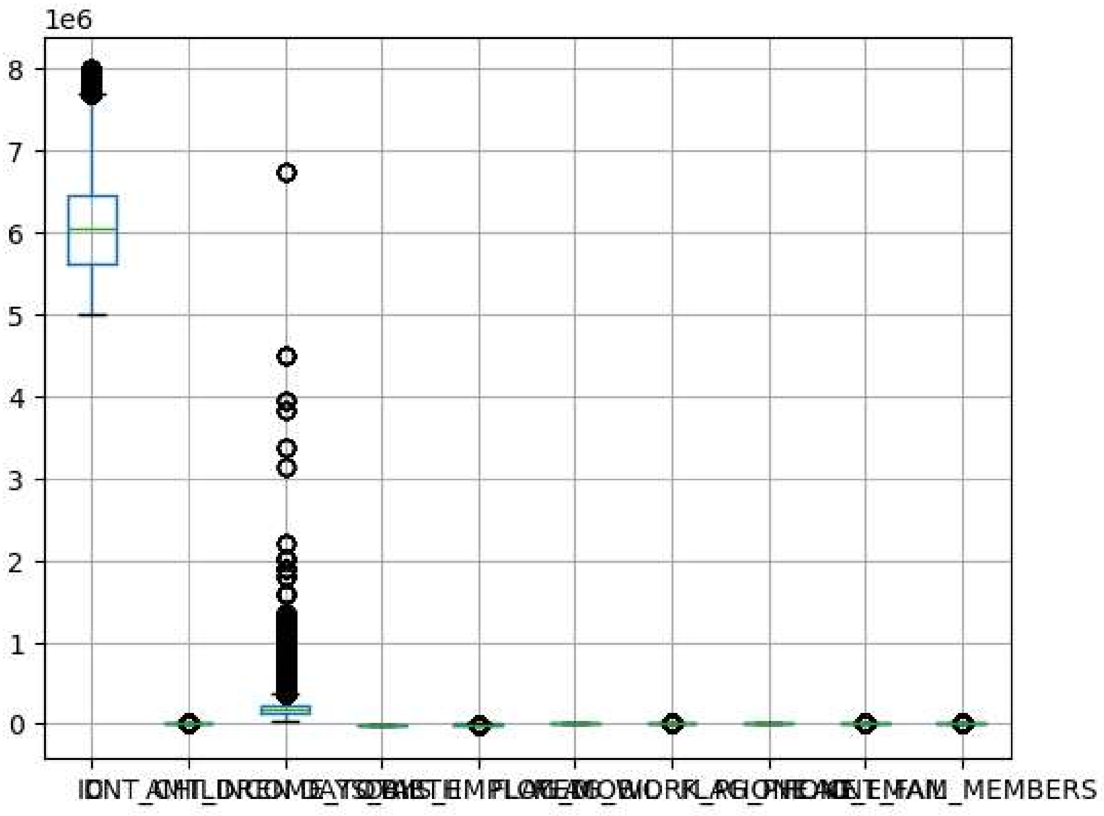
Out[9]:
```
ID                          0
CODE_GENDER                 0
FLAG_OWN_CAR                0
FLAG_OWN_REALTY             0
CNT_CHILDREN                0
AMT_INCOME_TOTAL            0
NAME_INCOME_TYPE            0
NAME_EDUCATION_TYPE         0
NAME_FAMILY_STATUS          0
NAME_HOUSING_TYPE           0
DAYS_BIRTH                  0
DAYS_EMPLOYED               0
FLAG_MOBIL                  0
FLAG_WORK_PHONE             0
FLAG_PHONE                  0
FLAG_EMAIL                  0
OCCUPATION_TYPE        134203
CNT_FAM_MEMBERS             0
STATUS                 402100
dtype: int64
```

In [4]:
```
df.dropna(subset = ['OCCUPATION_TYPE'],inplace = True , ignore_index= Tr
```

In [84]:
```
df.isnull().sum()
```

Out[84]:
```
ID                          0
CODE_GENDER                 0
FLAG_OWN_CAR                0
FLAG_OWN_REALTY             0
CNT_CHILDREN                0
AMT_INCOME_TOTAL            0
NAME_INCOME_TYPE            0
NAME_EDUCATION_TYPE         0
NAME_FAMILY_STATUS          0
NAME_HOUSING_TYPE           0
DAYS_BIRTH                  0
DAYS_EMPLOYED               0
FLAG_MOBIL                  0
FLAG_WORK_PHONE             0
FLAG_PHONE                  0
FLAG_EMAIL                  0
OCCUPATION_TYPE             0
CNT_FAM_MEMBERS             0
STATUS                 279220
dtype: int64
```

In [22]:
```
df.boxplot()
plt.show()
```



In [5]:
```
test = df[df['STATUS'].isna()]
```

In [6]:
```
train = df[df['STATUS'].notna()]
```

In [28]:
```
test.isnull().sum()
```

Out[28]:
```
ID                          0
CODE_GENDER                 0
FLAG_OWN_CAR                0
FLAG_OWN_REALTY             0
CNT_CHILDREN                0
AMT_INCOME_TOTAL            0
NAME_INCOME_TYPE            0
NAME_EDUCATION_TYPE         0
NAME_FAMILY_STATUS          0
NAME_HOUSING_TYPE           0
DAYS_BIRTH                  0
DAYS_EMPLOYED               0
FLAG_MOBIL                  0
FLAG_WORK_PHONE             0
FLAG_PHONE                  0
FLAG_EMAIL                  0
OCCUPATION_TYPE             0
CNT_FAM_MEMBERS             0
STATUS                 279220
dtype: int64
```

In [27]:
```
train.isnull().sum()
```

Out[27]:
```
ID                     0
CODE_GENDER            0
FLAG_OWN_CAR           0
FLAG_OWN_REALTY        0
CNT_CHILDREN           0
AMT_INCOME_TOTAL       0
NAME_INCOME_TYPE       0
NAME_EDUCATION_TYPE    0
NAME_FAMILY_STATUS     0
NAME_HOUSING_TYPE      0
DAYS_BIRTH             0
DAYS_EMPLOYED          0
FLAG_MOBIL             0
FLAG_WORK_PHONE        0
FLAG_PHONE             0
FLAG_EMAIL             0
OCCUPATION_TYPE        0
CNT_FAM_MEMBERS        0
STATUS                 0
dtype: int64
```

In [7]:
```
from sklearn.preprocessing import LabelEncoder
```

In [8]:
```
train.shape , test.shape
```

Out[8]:
```
((25134, 19), (279220, 19))
```

In [9]:
```
X_train = train.iloc[: , train.columns != 'STATUS']
y_train = train[['STATUS']]
```

In [10]:
```
X_test = test.iloc[: , test.columns != 'STATUS']
y_test = test[['STATUS']]
```

In [11]:
```
for col in X_train.columns:
    if X_train[col].dtypes == 'object':
        X_train[col] = LabelEncoder().fit_transform(X_train[col])
```

In [12]:
```
for col in X_test.columns:
    if X_test[col].dtypes == 'object':
        X_test[col] = LabelEncoder().fit_transform(X_test[col])
```

In [13]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score , GridSearchCV
from sklearn.metrics import accuracy_score
```

In [14]:
```
dtr = DecisionTreeClassifier(random_state=42)
param = {'criterion': ['gini' , 'entropy'] ,
        'max_depth': range(1,16)}
gscv = GridSearchCV(dtr , param , scoring= 'accuracy' , cv= 5)
gscv.fit(X_train , y_train)
```

Out[14]:
```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': range(1, 16)},
             scoring='accuracy')
```

**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 [15]:
```
dt = gscv.best_estimator_
```

In [16]:
```
dt
```

Out[16]:
```
DecisionTreeClassifier(max_depth=2, random_state=42)
```

**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 [17]:
```
gscv.feature_names_in_
```

Out[17]:
```
array(['ID', 'CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY',
       'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'NAME_INCOME_TYPE',
       'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYP
E',
       'DAYS_BIRTH', 'DAYS_EMPLOYED', 'FLAG_MOBIL', 'FLAG_WORK_PHONE',
       'FLAG_PHONE', 'FLAG_EMAIL', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBER
S'],
      dtype=object)
```

In [18]:
```
dt.fit(X_train , y_train)
score = cross_val_score(dt , X_train , y_train , scoring='accuracy' , c
pred = dt.predict(X_test)
```

In [19]:
```
score
```

Out[19]:
```
0.49013288449297787
```

In [20]:
```
ypred = dt.predict(X_train)
accuracy = accuracy_score(y_train , ypred)
```

In [21]:
```
accuracy
```

Out[21]:
```
0.4901328877218111
```

In [22]:
```
pred
```

Out[22]:
```
array(['C', 'C', 'C', ..., 'C', 'C', 'C'], dtype=object)
```

In [23]:
```
X_test.index
```

Out[23]:
```
Index([     7,     35,     36,     37,     53,     57,    123,    124,
125,
          149,
       ...
       304344, 304345, 304346, 304347, 304348, 304349, 304350, 304351,
304352,
       304353],
      dtype='int64', length=279220)
```

In [24]:
```
ts = pd.DataFrame(pred , index = X_test.index , columns = ['STATUS'])
```

In [25]:
```
ts.nunique()
```

Out[25]:
```
STATUS    1
dtype: int64
```

In [26]:
```
ts.value_counts()
```
Out[26]:
```
STATUS
C         279220
Name: count, dtype: int64
```

In [27]:
```
X_test
```
Out[27]:

| | ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AM |
|---|---|---|---|---|---|---|
| **7** | 6153651 | 1 | 1 | 1 | 0 | |
| **35** | 6153733 | 1 | 1 | 1 | 0 | |
| **36** | 6153734 | 1 | 1 | 1 | 0 | |
| **37** | 6153735 | 1 | 1 | 1 | 0 | |
| **53** | 6153736 | 0 | 1 | 1 | 2 | |
| **...** | ... | ... | ... | ... | ... | |
| **304349** | 6837707 | 1 | 0 | 1 | 0 | |
| **304350** | 6839936 | 1 | 1 | 1 | 1 | |
| **304351** | 6840222 | 0 | 0 | 0 | 0 | |
| **304352** | 6841878 | 0 | 0 | 0 | 0 | |
| **304353** | 6842885 | 0 | 0 | 1 | 0 | |

279220 rows × 18 columns

In [28]:
```
pd.merge(X_test , ts , how ='left' , left_index= True , right_index= Tru
```

Out[28]:

| | ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AM |
|---|---|---|---|---|---|---|
| **7** | 6153651 | 1 | 1 | 1 | 0 | |
| **35** | 6153733 | 1 | 1 | 1 | 0 | |
| **36** | 6153734 | 1 | 1 | 1 | 0 | |
| **37** | 6153735 | 1 | 1 | 1 | 0 | |
| **53** | 6153736 | 0 | 1 | 1 | 2 | |
| **...** | ... | ... | ... | ... | ... | |
| **304349** | 6837707 | 1 | 0 | 1 | 0 | |
| **304350** | 6839936 | 1 | 1 | 1 | 1 | |
| **304351** | 6840222 | 0 | 0 | 0 | 0 | |
| **304352** | 6841878 | 0 | 0 | 0 | 0 | |
| **304353** | 6842885 | 0 | 0 | 1 | 0 | |

279220 rows × 19 columns

In [29]:
```
df['STATUS'].fillna('C',inplace = True)
```

In [30]:
```
df.isnull().sum()
```

Out[30]:
```
ID                     0
CODE_GENDER            0
FLAG_OWN_CAR           0
FLAG_OWN_REALTY        0
CNT_CHILDREN           0
AMT_INCOME_TOTAL       0
NAME_INCOME_TYPE       0
NAME_EDUCATION_TYPE    0
NAME_FAMILY_STATUS     0
NAME_HOUSING_TYPE      0
DAYS_BIRTH             0
DAYS_EMPLOYED          0
FLAG_MOBIL             0
FLAG_WORK_PHONE        0
FLAG_PHONE             0
FLAG_EMAIL             0
OCCUPATION_TYPE        0
CNT_FAM_MEMBERS        0
STATUS                 0
dtype: int64
```

In [31]:
```
df.shape
```

Out[31]:
```
(304354, 19)
```

In [34]:
```
df.dtypes
```

Out[34]:
```
ID                       int64
CODE_GENDER             object
FLAG_OWN_CAR            object
FLAG_OWN_REALTY         object
CNT_CHILDREN             int64
AMT_INCOME_TOTAL       float64
NAME_INCOME_TYPE        object
NAME_EDUCATION_TYPE     object
NAME_FAMILY_STATUS      object
NAME_HOUSING_TYPE       object
DAYS_BIRTH               int64
DAYS_EMPLOYED            int64
FLAG_MOBIL               int64
FLAG_WORK_PHONE          int64
FLAG_PHONE               int64
FLAG_EMAIL               int64
OCCUPATION_TYPE         object
CNT_FAM_MEMBERS          int64
STATUS                  object
dtype: object
```

In [29]:
```
df.head()
```

Out[29]:

| | ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AMT_INC |
|---|---|---|---|---|---|---|
| **0** | 5008806 | M | Y | Y | 0 | |
| **1** | 5008808 | F | N | Y | 0 | |
| **2** | 5008809 | F | N | Y | 0 | |
| **3** | 5008810 | F | N | Y | 0 | |
| **4** | 5008811 | F | N | Y | 0 | |

In [43]:
```
df.isnull().sum()
```

Out[43]:
```
ID                     0
CODE_GENDER            0
FLAG_OWN_CAR           0
FLAG_OWN_REALTY        0
CNT_CHILDREN           0
AMT_INCOME_TOTAL       0
NAME_INCOME_TYPE       0
NAME_EDUCATION_TYPE    0
NAME_FAMILY_STATUS     0
NAME_HOUSING_TYPE      0
DAYS_BIRTH             0
DAYS_EMPLOYED          0
FLAG_MOBIL             0
FLAG_WORK_PHONE        0
FLAG_PHONE             0
FLAG_EMAIL             0
OCCUPATION_TYPE        0
CNT_FAM_MEMBERS        0
STATUS                 0
dtype: int64
```

In [32]:
```
df1 = df.copy()
```

In [33]:
```
df['TARGET'] = 'NAN'
```

In [34]:
```
df['TARGET'] = np.where(df['STATUS'] == 'X' , 'approve' , df['TARGET'])
```

In [35]:
```
df['TARGET'] = np.where(df['STATUS'] == 'C' , 'approve' , df['TARGET'])
```

In [36]:
```
df['TARGET'] = np.where(df['STATUS'] == '0' , 'approve' , df['TARGET'])
```

In [37]:
```
df['TARGET'] = np.where(df['STATUS'] == '1' , 'notapprove' , df['TARGET
```

In [38]:
```
df['TARGET'] = np.where(df['STATUS'] == '2' , 'notapprove' , df['TARGET
```

In [39]:
```
df['TARGET'] = np.where(df['STATUS'] == '3' , 'notapprove' , df['TARGET
```

In [40]:
```
df['TARGET'] = np.where(df['STATUS'] == '4' , 'notapprove' , df['TARGET
```

In [41]:
```
df['TARGET'] = np.where(df['STATUS'] == '5' , 'notapprove' , df['TARGET
```

In [42]:
```
df['TARGET'] = np.where(df['STATUS'] == '6' , 'notapprove' , df['TARGET
```

In [43]:
```
df.head()
```

Out[43]:

| | ID | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AMT_INC |
|---|---|---|---|---|---|---|
| **0** | 5008806 | M | Y | Y | 0 | |
| **1** | 5008808 | F | N | Y | 0 | |
| **2** | 5008809 | F | N | Y | 0 | |
| **3** | 5008810 | F | N | Y | 0 | |
| **4** | 5008811 | F | N | Y | 0 | |

In [44]:
```
df.dtypes
```

Out[44]:
```
ID                       int64
CODE_GENDER             object
FLAG_OWN_CAR            object
FLAG_OWN_REALTY         object
CNT_CHILDREN             int64
AMT_INCOME_TOTAL       float64
NAME_INCOME_TYPE        object
NAME_EDUCATION_TYPE     object
NAME_FAMILY_STATUS      object
NAME_HOUSING_TYPE       object
DAYS_BIRTH               int64
DAYS_EMPLOYED            int64
FLAG_MOBIL               int64
FLAG_WORK_PHONE          int64
FLAG_PHONE               int64
FLAG_EMAIL               int64
OCCUPATION_TYPE         object
CNT_FAM_MEMBERS          int64
STATUS                  object
TARGET                  object
dtype: object
```

In [45]:
```
df['STATUS'] = df['STATUS'].astype('str')
```

In [46]:
```
df.drop(['CODE_GENDER'],axis = 1 , inplace= True)
```

In [48]:
```
X = df.iloc[: , df.columns != 'TARGET']
y = df[['TARGET']]
```

In [49]:
```
for col in X.columns:
    if X[col].dtypes == 'object':
        X[col] = LabelEncoder().fit_transform(X[col])
```

In [50]:
```
y.value_counts()
```

Out[50]:
```
TARGET
approve       304073
notapprove       281
Name: count, dtype: int64
```

In [51]:
```
from imblearn.over_sampling import SMOTE
```

In [52]:
```
X_re , y_re = SMOTE(random_state=42).fit_resample(X , y)
```

In [53]:
```
y_re.value_counts()
```

Out[53]:
```
TARGET
approve       304073
notapprove    304073
Name: count, dtype: int64
```

In [54]:
```
from sklearn.model_selection import train_test_split , GridSearchCV , c
```

In [55]:
```
X_train , X_test , y_train , y_test =train_test_split(X_re , y_re , tes
```

In [56]:
```
from sklearn.ensemble import AdaBoostClassifier , GradientBoostingClass
```

In [ ]:
```
ac = AdaBoostClassifier()
param = {'n_estimators': [20 , 30 , 70] ,
        'learning_rate':[0.001, 0.5 , 0.1,1,10,100,0.8]}
gscv = GridSearchCV(ac , param , scoring='accuracy' , cv = 5 )
gscv.fit(X_train , y_train)
```

In [57]:
```
ac = AdaBoostClassifier()
```

In [58]:
```
ac.fit(X_train , y_train)
```

Out[58]:
```
AdaBoostClassifier()
```

**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 [59]:
```
yp = ac.predict(X_test)
```

In [60]:
```
accuracy = accuracy_score(y_test , yp)
```

In [61]:
```
accuracy
```

Out[61]:
```
1.0
```

In [62]:
```
ga = GradientBoostingClassifier()
```

In [63]:
```
ga.fit(X_train , y_train)
```

Out[63]:
```
GradientBoostingClassifier()
```

**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 [64]:
```
ypg = ga.predict(X_test)
```

In [65]:
```
accuracyg = accuracy_score(y_test , ypg)
accuracyg
```

Out[65]:
```
1.0
```

In [ ]:
```
## by using voting classifier
```

In [66]:
```
from sklearn.ensemble import VotingClassifier
```

In [67]:
```
vc = VotingClassifier([('dt' , DecisionTreeClassifier()) ,
                       ('ab' , AdaBoostClassifier()) ,
                       ('gb' , GradientBoostingClassifier())])
```

In [68]:
```
vc.fit(X_train , y_train)
```

Out[68]:
```
VotingClassifier(estimators=[('dt', DecisionTreeClassifier()),
                             ('ab', AdaBoostClassifier()),
                             ('gb', GradientBoostingClassifier())])
```

**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 [69]:
```
vp = vc.predict(X_test)
```

In [70]:
```
accuracy = accuracy_score(y_test , vp)
```

In [71]:
```
accuracy
```
Out[71]:
```
1.0
```

In [72]:
```
vpp = pd.DataFrame(vp)
```

In [73]:
```
vpp.value_counts()
```
Out[73]:
```
approve       121978
notapprove    121281
Name: count, dtype: int64
```

In [74]:
```
from sklearn.metrics import classification_report
```

In [75]:
```
cm = classification_report(y_test , vp)
```

In [76]:
```
print(cm)
```
```
              precision    recall  f1-score   support

     approve       1.00      1.00      1.00    121978
  notapprove       1.00      1.00      1.00    121281

    accuracy                           1.00    243259
   macro avg       1.00      1.00      1.00    243259
weighted avg       1.00      1.00      1.00    243259
```

In [ ]: