Author : Dimgba Martha Otisi

@martha_samuel_

score = 0.8308659

In [ ]:

```
import pandas as pd
pd.set_option('display.max_columns', None)
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split, GridSearchCV, Stra
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import roc_curve, roc_auc_score, log_loss
from sklearn.metrics import classification_report
from sklearn.ensemble import GradientBoostingClassifier
from matplotlib import pyplot
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
from xgboost import XGBClassifier
import catboost
from catboost import CatBoostClassifier
import lightgbm as lgb
from lightgbm import LGBMClassifier
from sklearn.metrics import mean_squared_error
from statsmodels.stats.outliers_influence import variance_inflation_fact
from sklearn.feature_selection import RFE, RFECV
from sklearn.model_selection import cross_val_score, KFold
```

In [ ]:

```
df= pd.read_csv('Train1.csv')
df_Test = pd.read_csv('Test1.csv')
#print(df.head(3))
df_sample=pd.read_csv("SampleSubmission1.csv")
```

In [ ]:

```
df_user = df.pop('Applicant_ID')
df_Test_user =df_Test.pop('Applicant_ID')
```

In [ ]:

```
# combine both test and train to
df_Test['default_status']=-1

data = pd.concat((df, df_Test)).reset_index(drop=True)
print(df.shape, df_Test.shape, data.shape)
data.head()
```

In [ ]:

```
data.info()
```

In [ ]:

```
data.isna().sum()#checks for number of missing data
```

In [ ]:

```
#mapping and converting categoricals
data['form_field47']= data['form_field47'].map({'charge':0, 'lending':1}

data['default_status']= data['default_status'].map({'yes':1, 'no':0}, na
```

In [ ]:

```
#data.describe()
data['form_field36'].mean()
```

In [ ]:

```
np.all(np.isfinite(data))#this shows there is infinity
```

In [ ]:

```

```

In [ ]:

```
data.columns
```

feature engineering

In [ ]:

```
data.form_field17=data.form_field17.combine_first(data.form_field18)#it
#print(df.form_field18)
data.form_field17.isna().sum()
#df['form_field17'].fillna(method='backfill', inplace=True, axis=1)#take
#print(df['form_field17'])
```

In [ ]:

```
#data['form_field18'].fillna(method='ffill', inplace=True)#takes values
#print(data['form_field18'])

data.form_field18=data.form_field18.combine_first(data.form_field17)#it
#print(data.form_field18)
data.form_field18.isna().sum()
#or data.form_field18=data.form_field18.fillna(data.form_field17)
#print(sum(data.form_field18))
#print(data.form_field18)
```

In [ ]:

```
data.form_field20=data.form_field20.combine_first(data.form_field19)#it
#print(data_Test.form_field20)
data.form_field20.isna().sum()

#df['form_field20'].fillna(method='ffill', inplace=True, axis=1)# fills
#print(df['form_field20'])
```

In [ ]:

```
data.form_field19=data.form_field19.combine_first(data.form_field20)#it
#print(data.form_field18)
data.form_field19.isna().sum()
#['form_field19'].fillna(method='backfill', inplace=True, axis=1)  # fi
#print(data['form_field19']) or print(data.form_field19)
```

In [ ]:

In [ ]:

```
# Function to calculate missing values by column# Funct
def missing_values_table(df):
        # Total missing values
        mis_val = data.isnull().sum()

        # Percentage of missing values
        mis_val_percent = 100 * data.isnull().sum() / len(data)

        # Make a table with the results
        mis_val_table = pd.concat([mis_val, mis_val_percent], axis=1)

        # Rename the columns
        mis_val_table_ren_columns = mis_val_table.rename(
        columns = {0 : 'Missing Values', 1 : '% of Total Values'})

        # Sort the table by percentage of missing descending
        mis_val_table_ren_columns = mis_val_table_ren_columns[
            mis_val_table_ren_columns.iloc[:,1] != 0].sort_values(
        '% of Total Values', ascending=False).round(1)

        # Print some summary information
        print ("Your selected dataframe has " + str(data.shape[1]) + " c
            "There are " + str(mis_val_table_ren_columns.shape[0]) +
              " columns that have missing values.")

        # Return the dataframe with missing information
        return mis_val_table_ren_columns
```

In [ ]:

```
# Missing values statistics
missing_values = missing_values_table(data)
missing_values.head(20)
```

In [ ]:

```
data=data.drop(['form_field40'],axis=1)# at here,with feat.eng with -1=(
```

In [ ]:

```
data.describe()
```

In [ ]:

```
# this replaces Nan with -1
data=data.fillna(-1)
```

In [ ]:

```
dd=data.copy()
dd.drop('default_status', axis=1, inplace=True)

ddc = dd.copy()

dd.shape,ddc.shape
```

In [ ]:

```
#Checking for multi colinearity
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dd.values, i) for i in ra
vif["features"] = dd.columns
vif
```

In [454]:

```
#Scaling/normalizing the dataset
dd = StandardScaler().fit_transform(dd)
```

In [455]:

```
#Checking for multi colinearity
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dd, i) for i in range(dd.
vif["features"] = ddc.columns
vif
```

Out[455]:

	VIF Factor	features
0	1.175753	form_field2
1	3.239841	form_field3
2	3.701381	form_field5
3	1.514017	form_field6
4	2.381217	form_field8
5	2.822712	form_field9
6	1.167424	form_field11
7	1.674684	form_field12
8	1.249478	form_field13
9	1.000228	form_field14
10	1.085605	form_field15
11	2.400408	form_field16

In [456]:

```
corr=ddc.corr()
corr.style.background_gradient(cmap='coolwarm')
```

Out[456]:

	form_field2	form_field3	form_field5	form_field6	form_field8	form_field9
form_field2	1.000000	0.120972	0.039859	-0.145104	-0.177353	-0.147461
form_field3	0.120972	1.000000	0.658141	-0.127995	-0.193187	-0.191957
form_field5	0.039859	0.658141	1.000000	-0.050649	-0.074270	-0.082461
form_field6	-0.145104	-0.127995	-0.050649	1.000000	0.508145	0.262408
form_field8	-0.177353	-0.193187	-0.074270	0.508145	1.000000	0.571441
form_field9	-0.147461	-0.191957	-0.082461	0.262408	0.571441	1.000000
form_field11	0.053257	0.309079	0.220126	-0.042488	-0.066809	-0.068259
form_field12	-0.020493	-0.145556	-0.064107	-0.024221	0.210455	0.335414
form_field13	-0.061538	-0.097108	-0.041852	0.178367	0.327796	0.254227
form_field14	-0.003896	-0.001201	-0.002439	-0.000507	0.000132	-0.000367
form_field15	-0.062963	-0.089337	-0.050925	0.129650	0.216510	0.165126

In [14]:

```
data['form_field17_19'] = data['form_field17'] + data['form_field19']
data =data.drop(['form_field17','form_field19'],axis=1)
```

In [15]:

```
data['form_field18_20'] = (data['form_field18'] + data['form_field20'])
data=data.drop(['form_field18','form_field20'],axis=1)
```

In [16]:

```
data['form_field1719_1820'] = (data['form_field17_19'] + data['form_fiel
data=data.drop(['form_field17_19','form_field18_20'],axis=1)
```

In [17]:

```
data['form_field32_37'] = (data['form_field32'] + data['form_field37'])
data=data.drop(['form_field32','form_field37'],axis=1)
```

In [18]:

```
data['form_field4_46'] = data['form_field4'] + data['form_field46']
data=data.drop(['form_field4','form_field46'],axis=1)
```

In [19]:

```
data['form_field1_28'] = data['form_field1'] + data['form_field28']
data =data.drop(['form_field1','form_field28'],axis=1)
```

In [20]:

```
data['form_field26_27'] = (data['form_field26'] +  data['form_field27'])
data=data.drop(['form_field26','form_field27'],axis=1)
```

In [21]:

```
data['form_field7_10'] = (data['form_field7'] + data['form_field10'])
data=data.drop(['form_field7','form_field10'],axis=1)
```

In [22]:

```
data['form_field25_29'] = (data['form_field25'] + data['form_field29'])
data=data.drop(['form_field25','form_field29'],axis=1)
```

In [23]:
```
data=data.drop(['form_field1719_1820'],axis=1) # at thisnew point,missi
```

In [457]:
```
data=data.replace([np.inf, -np.inf], np.nan)

data=data.fillna(-1)
```

In [132]:
```
print(data.shape)
data.head(1)
data.columns
```

```
(80000, 39)
```

Out[132]:
```
Index(['form_field2', 'form_field3', 'form_field5', 'form_field6',
       'form_field8', 'form_field9', 'form_field11', 'form_field12',
       'form_field13', 'form_field14', 'form_field15', 'form_field16',
       'form_field21', 'form_field22', 'form_field23', 'form_field24',
       'form_field30', 'form_field33', 'form_field34', 'form_field35',
       'form_field36', 'form_field38', 'form_field39', 'form_field41',
       'form_field42', 'form_field43', 'form_field44', 'form_field45',
       'form_field47', 'form_field48', 'form_field49', 'form_field50',
       'default_status', 'form_field32_37', 'form_field4_46', 'form_fi
eld1_28',
       'form_field26_27', 'form_field7_10', 'form_field25_29'],
      dtype='object')
```

In [ ]:

model

In [24]:

```

X=data[:df.shape[0]]
X.drop('default_status', axis=1, inplace=True)
X = StandardScaler().fit_transform(X)

y = data.default_status[:df.shape[0]].copy()#map({'yes':1, 'no':0}, na_a


print(len(X),len(y))
print(X.shape[1])#the no of columns
print(y.tail())
#print(X.head())
```

```
/home/martha/anaconda3/lib/python3.7/site-packages/pandas/core/frame.p
y:3997: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  errors=errors,

56000 56000
39
55995    0.0
55996    1.0
55997    0.0
55998    0.0
55999    0.0
Name: default_status, dtype: float64
```

In [25]:

```
x=data[df.shape[0]:].astype(float)
x.drop('default_status', axis=1,inplace=True)
x = StandardScaler().fit_transform(x)
print(len(x))
```

```
24000
```

In [26]:

```
#Split the data into 80% training and 20% testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =0.2
```

In [ ]:

```
# this and the next 7 cells were run in colab

pip install scikit-optimize

```

```
Collecting scikit-optimize Downloading
https://files.pythonhosted.org/packages/8b/03/be33e89f55866065a02e515c5b319304a801a9f102
0.8.1-py2.py3-none-any.whl
(https://files.pythonhosted.org/packages/8b/03/be33e89f55866065a02e515c5b319304a801a9f10
0.8.1-py2.py3-none-any.whl) (101kB) |████████████████████████████████|
102kB 662kB/s Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.6/dist-
packages (from scikit-optimize) (1.18.5) Requirement already satisfied: scikit-learn>=0.20.0 in
```

```
/usr/local/lib/python3.6/dist-packages (from scikit-optimize) (0.22.2.post1) Collecting
pyaml>=16.9 Downloading
https://files.pythonhosted.org/packages/15/c4/1310a054d33abc318426a956e7d6df0df76a6ddfa9c
20.4.0-py2.py3-none-any.whl
(https://files.pythonhosted.org/packages/15/c4/1310a054d33abc318426a956e7d6df0df76a6ddfa9
20.4.0-py2.py3-none-any.whl) Requirement already satisfied: scipy>=0.19.1 in
/usr/local/lib/python3.6/dist-packages (from scikit-optimize) (1.4.1) Requirement already
satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-packages (from scikit-optimize) (0.16.0)
Requirement already satisfied: PyYAML in /usr/local/lib/python3.6/dist-packages (from
pyaml>=16.9->scikit-optimize) (3.13) Installing collected packages: pyaml, scikit-optimize
Successfully installed pyaml-20.4.0 scikit-optimize-0.8.1
```

In [ ]:

```
pip install shap
```

```
Collecting shap Downloading
https://files.pythonhosted.org/packages/d2/17/37ee6c79cafbd9bb7423b54e55ea90beec66aa7638
0.36.0.tar.gz
(https://files.pythonhosted.org/packages/d2/17/37ee6c79cafbd9bb7423b54e55ea90beec66aa763
0.36.0.tar.gz) (319kB) |████████████████████████████████| 327kB 2.6MB/s
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from shap)
(1.18.5) Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from
shap) (1.4.1) Requirement already satisfied: scikit-learn in /usr/local/lib/python3.6/dist-packages
(from shap) (0.22.2.post1) Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-
packages (from shap) (1.1.2) Requirement already satisfied: tqdm>4.25.0 in
/usr/local/lib/python3.6/dist-packages (from shap) (4.41.1) Collecting slicer Downloading
https://files.pythonhosted.org/packages/46/cf/f37ac7f61214ed044b0df91252ab19376de5587926c
0.0.4-py3-none-any.whl
(https://files.pythonhosted.org/packages/46/cf/f37ac7f61214ed044b0df91252ab19376de5587926
0.0.4-py3-none-any.whl) Requirement already satisfied: numba in /usr/local/lib/python3.6/dist-
packages (from shap) (0.48.0) Requirement already satisfied: joblib>=0.11 in
/usr/local/lib/python3.6/dist-packages (from scikit-learn->shap) (0.16.0) Requirement already
satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas->shap) (2018.9)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.6/dist-packages
(from pandas->shap) (2.8.1) Requirement already satisfied: llvmlite<0.32.0,>=0.31.0dev0 in
/usr/local/lib/python3.6/dist-packages (from numba->shap) (0.31.0) Requirement already
satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from numba->shap) (50.3.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-
dateutil>=2.7.3->pandas->shap) (1.15.0) Building wheels for collected packages: shap Building
wheel for shap (setup.py) ... done Created wheel for shap: filename=shap-0.36.0-cp36-cp36m-
linux_x86_64.whl size=456466
sha256=54c463fb2ccd2e6e5ac530cc9b78a9bff1f09ba7cded994effeca711399f743e Stored in
directory:
/root/.cache/pip/wheels/fb/15/e1/8f61106790da27e0765aaa6e664550ca2c50ea339099e799f4
Successfully built shap Installing collected packages: slicer, shap Successfully installed shap-
0.36.0 slicer-0.0.4
```

In [ ]:

```
from skopt import gp_minimize
from skopt.space import Real, Integer
from skopt.utils import use_named_args
from skopt.plots import plot_convergence
from copy import deepcopy
import pprint
import shap
pp = pprint.PrettyPrinter(indent=4)
% matplotlib inline
```

In [ ]:

```
class ModelOptimizer:
    best_score = None
    opt = None

    def __init__(self, model, X_train, y_train, categorical_columns_ind
        self.model = model
        self.X_train = X_train
        self.y_train = y_train
        self.categorical_columns_indices = categorical_columns_indices
        self.n_fold = n_fold
        self.seed = seed
        self.early_stopping_rounds = early_stopping_rounds
        self.is_stratified = is_stratified
        self.is_shuffle = is_shuffle


    def update_model(self, **kwargs):
        for k, v in kwargs.items():
            setattr(self.model, k, v)

    def evaluate_model(self):
        pass

    def optimize(self, param_space, max_evals=10, n_random_starts=2):
        start_time = time.time()

        @use_named_args(param_space)
        def _minimize(**params):
            self.model.set_params(**params)
            return self.evaluate_model()

        opt = gp_minimize(_minimize, param_space, n_calls=max_evals, n_r
        best_values = opt.x
        optimal_values = dict(zip([param.name for param in param_space],
        best_score = opt.fun
        self.best_score = best_score
        self.opt = opt

        print('optimal_parameters: {}\noptimal score: {}\noptimization t
        print('updating model with optimal values')
        self.update_model(**optimal_values)
        plot_convergence(opt)
        return optimal_values


class CatboostOptimizer(ModelOptimizer):
    def evaluate_model(self):
        validation_scores = catboost.cv(
        catboost.Pool(self.X_train,
                      self.y_train,
                      cat_features=self.categorical_columns_indices),
                      self.model.get_params(),
                      nfold=self.n_fold,
                      stratified=self.is_stratified,
                      seed=self.seed,
                      early_stopping_rounds=self.early_stopping_rounds,
                             shuffle=self.is_shuffle,
                             plot=False)

        self.scores = validation_scores
        test_scores = validation_scores.iloc[:, 1]
        best_metric = test_scores.max()
        return 1 - best_metric
```

In [ ]:

```
# default param for catboost
default_cb = catboost.CatBoostClassifier(loss_function='Logloss', eval_m
default_cb_optimizer = CatboostOptimizer(default_cb, X_train, y_train)
default_cb_optimizer.evaluate_model()
```

```
Stopped by overfitting detector (30 iterations wait) 0.16170255920956234
```

In [ ]:

```
import time
#  greedy parameter tuning
cb = catboost.CatBoostClassifier(n_estimators=4000, # use large n_estima
                                 one_hot_max_size=2,
                                 loss_function='Logloss',
                                 eval_metric='AUC',
                                 boosting_type='Ordered', # use permutations
                                 random_seed=2405,
                                 use_best_model=True,
                                 silent=True)
cb_optimizer = CatboostOptimizer(cb, X_train, y_train)
params_space = [Real(0.01, 0.8, name='learning_rate'),]
cb_optimal_values = cb_optimizer.optimize(params_space)
```

```
Stopped by overfitting detector (30 iterations wait) Stopped by overfitting detector (30 iterations
wait) Stopped by overfitting detector (30 iterations wait) Stopped by overfitting detector (30
iterations wait) Stopped by overfitting detector (30 iterations wait) The objective has been
evaluated at this point before. Stopped by overfitting detector (30 iterations wait) The objective
has been evaluated at this point before. Stopped by overfitting detector (30 iterations wait)
Stopped by overfitting detector (30 iterations wait) The objective has been evaluated at this
point before. Stopped by overfitting detector (30 iterations wait) Stopped by overfitting detector
(30 iterations wait) optimal_parameters: {'learning_rate': 0.010074100733272565} optimal
score: 0.16163254028292506 optimization time: 4692.2240607738495 updating model with
optimal values
```

In [ ]:

```
params_space = [Integer(2, 10, name='max_depth'),]
cb_optimal_values = cb_optimizer.optimize(params_space)
```

```
Stopped by overfitting detector (30 iterations wait) Stopped by overfitting detector (30 iterations
wait) The objective has been evaluated at this point before. Stopped by overfitting detector (30
```

iterations wait) The objective has been evaluated at this point before. Stopped by overfitting detector (30 iterations wait) The objective has been evaluated at this point before. The objective has been evaluated at this point before. Stopped by overfitting detector (30 iterations wait) The objective has been evaluated at this point before. The objective has been evaluated at this point before. Stopped by overfitting detector (30 iterations wait) The objective has been evaluated at this point before. optimal_parameters: {'max_depth': 10} optimal score: 0.16150794977413196 optimization time: 22162.268199443817 updating model with optimal values

In [ ]:

```
params_space = [
                Real(0.5, 1.0, name='colsample_bylevel'),
                Real(0.0, 100, name='bagging_temperature'),]
cb_optimal_values = cb_optimizer.optimize(params_space)
```

Stopped by overfitting detector (30 iterations wait) The objective has been evaluated at this point before. The objective has been evaluated at this point before. optimal_parameters: {'colsample_bylevel': 0.5, 'bagging_temperature': 100.0} optimal score: 0.1632450618509852 optimization time: 7268.540562152863 updating model with optimal values

In [ ]:

In [42]:

```
ctb = CatBoostClassifier(bagging_temperature=100.0,colsample_bylevel=0.5
                         learning_rate=0.010074100733272565,
                        max_depth=10,n_estimators=4000)

ctb.fit(X_train, y_train)
#ctb.fit(X, y)
```

```
0:      total: 133ms    remaining: 8m 52s
1:      total: 264ms    remaining: 8m 47s
2:      total: 417ms    remaining: 9m 15s
3:      total: 543ms    remaining: 9m 2s
4:      total: 686ms    remaining: 9m 7s
5:      total: 847ms    remaining: 9m 23s
6:      total: 991ms    remaining: 9m 25s
7:      total: 1.13s    remaining: 9m 23s
8:      total: 1.28s    remaining: 9m 27s
9:      total: 1.42s    remaining: 9m 24s
10:     total: 1.57s    remaining: 9m 29s
11:     total: 1.7s     remaining: 9m 24s
12:     total: 1.87s    remaining: 9m 32s
13:     total: 2s       remaining: 9m 30s
14:     total: 2.16s    remaining: 9m 33s
15:     total: 2.31s    remaining: 9m 35s
16:     total: 2.47s    remaining: 9m 37s
17:     total: 2.61s    remaining: 9m 37s
18:     total: 2.78s    remaining: 9m 42s
19:     total: 2.93s    remaining: 9m 43s
```

In [40]:

```
# evaluating the model
pred = ctb.predict(X_test)
#predict probabilities. keep only probabilities for positive outcome
pred_proba=ctb.predict_proba(X_test)[:, 1]

#printing the predictions
print(pred)
print(pred_proba)
print(len(pred))
```

```
[0. 1. 0. ... 0. 0. 0.]
[0.12612764 0.52493583 0.12786633 ... 0.3561645  0.17104266 0.0637060
1]
11200
```

Auc

In [41]:

```
#calculate scores. 100% correct prediction has auc score of 1
print('Auc on test set: {:.4f}'.format(roc_auc_score(y_test, pred_proba)
```

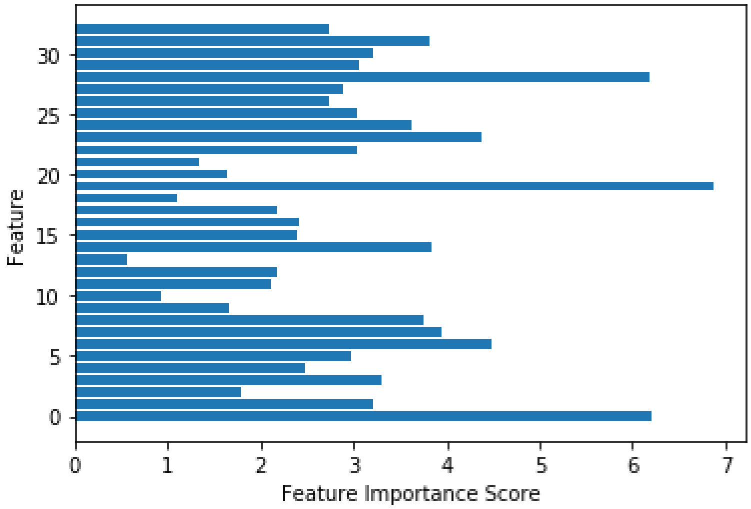
```
Auc on test set: 0.8355
```

In [ ]:

In [64]:

```
def plot_feature_importances_X(model):
    n_features = X.shape[1]
    plt.barh(range(n_features), model.feature_importances_, align='cente
    plt.yticks=(np.arange(n_features), df.columns)
    plt.xlabel('Feature Importance Score')
    plt.ylabel('Feature')

plot_feature_importances_X(ctb)
```



In [43]:

```
'''for pickle'''
with open ('MowopeMart1rfnow.pickle','wb') as f:
    pickle.dump(ctb, f)
```

In [47]:

```
'''we use this cell while testing on new data after we have written pick
'''to read the pickle'''
pickle_in = open('MowopeMart1rfnow.pickle','rb')
'''we renamed classifier here'''
ctb = pickle.load(pickle_in)
```

In [45]:

```
#df_sol = pd.DataFrame(ctb.predict(x))#converting prediction to a datafr
df_sol=ctb.predict_proba(x)[:, 1]
print(df_sol)
print(len(df_sol))
#print(df_sol_proba)
```

```
[0.16838516 0.48258375 0.28568059 ... 0.39304259 0.63251598 0.2459629
3]
24000
```

In [46]:

```
df_sample2=df_sample.copy()
df_sample2['default_status']=df_sol
df_sample2.to_csv('MowopeMart1rfnow.csv', index=False)
```

In [ ]:

```

```