Author : Dimgba Martha Otisi

@martha_samuel_

score = 0.8308659

In [2]:
```
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 [3]:
```
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 [4]:
```
df_user = df.pop('Applicant_ID')
df_Test_user =df_Test.pop('Applicant_ID')
```

In [5]:
```
# 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()
```

```
(56000, 51) (24000, 51) (80000, 51)
```

In [6]:
```
#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 [7]:
```
np.all(np.isfinite(data))#this shows there is infinity
```

Out[7]:
```
False
```

In [7]:
```
data.columns
```

Out[7]:
```
Index(['form_field1', 'form_field2', 'form_field3', 'form_field4',
       'form_field5', 'form_field6', 'form_field7', 'form_field8',
       'form_field9', 'form_field10', 'form_field11', 'form_field12',
       'form_field13', 'form_field14', 'form_field15', 'form_field16',
       'form_field17', 'form_field18', 'form_field19', 'form_field20',
       'form_field21', 'form_field22', 'form_field23', 'form_field24',
       'form_field25', 'form_field26', 'form_field27', 'form_field28',
       'form_field29', 'form_field30', 'form_field31', 'form_field32',
       'form_field33', 'form_field34', 'form_field35', 'form_field36',
       'form_field37', 'form_field38', 'form_field39', 'form_field40',
       'form_field41', 'form_field42', 'form_field43', 'form_field44',
       'form_field45', 'form_field46', 'form_field47', 'form_field48',
       'form_field49', 'form_field50', 'default_status'],
      dtype='object')
```

feature engineering

In [7]:
```
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'])
```

Out[7]:
```
14771
```

In [8]:
```
#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)
```

Out[8]:
```
14771
```

In [9]:
```
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'])
```

Out[9]: 4

In [10]:
```
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)
```

Out[10]: 4

In [ ]:
```
```

In [12]:
```
# 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 [13]:
```
# Missing values statistics
missing_values = missing_values_table(data)
missing_values.head(20)
```

```
Your selected dataframe has 51 columns.
There are 49 columns that have missing values.
```

Out[13]:

	Missing Values	% of Total Values
form_field40	62557	78.2
form_field31	56218	70.3
form_field41	54578	68.2
form_field45	44855	56.1
form_field11	44819	56.0
form_field23	40248	50.3
form_field30	36417	45.5
form_field35	33014	41.3
form_field15	32067	40.1
form_field48	29811	37.3
form_field22	29124	36.4
default_status	24000	30.0
form_field46	22789	28.5
form_field21	22561	28.2
form_field8	18964	23.7
form_field6	18964	23.7
form_field24	18902	23.6
form_field16	18510	23.1
form_field50	15853	19.8
form_field18	14771	18.5

In [11]:
```
data=data.drop(['form_field40'],axis=1)# at here,with feat.eng with -1=0
```

In [12]:
```
# this replaces Nan with -1
data=data.fillna(-1)
```

In [13]:
```
dd=data.copy()
dd.drop('default_status', axis=1, inplace=True)

ddc = dd.copy()

dd.shape,ddc.shape
```

Out[13]:
```
((80000, 49), (80000, 49))
```

In [17]:
```
#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
```

Out[17]:

	VIF Factor	features
0	18.132018	form_field1
1	1.557357	form_field2
2	4.164376	form_field3
3	2.111360	form_field4
4	3.981984	form_field5
5	1.792006	form_field6
6	6.989152	form_field7
7	3.200137	form_field8
8	4.600650	form_field9
9	9.965872	form_field10
10	1.296976	form_field11
11	2.273580	form_field12
12	1.381371	form_field13
13	1.001142	form_field14
14	1.532123	form_field15
15	8.006338	form_field16
16	99.383570	form_field17
17	94.966796	form_field18
18	61.084417	form_field19
19	59.641055	form_field20
20	5.558153	form_field21
21	5.341202	form_field22
22	3.560913	form_field23
23	1.110342	form_field24
24	8.462943	form_field25
25	14.226296	form_field26
26	14.232908	form_field27
27	2.342656	form_field28
28	10.259779	form_field29
29	2.371346	form_field30
30	1.660835	form_field31

	VIF Factor	features
31	15.352878	form_field32
32	2.302996	form_field33
33	1.995865	form_field34
34	1.827829	form_field35
35	6.716885	form_field36
36	9.468048	form_field37
37	4.710382	form_field38
38	2.098941	form_field39
39	1.744083	form_field41
40	2.820363	form_field42
41	7.376009	form_field43
42	3.094355	form_field44
43	2.454332	form_field45
44	2.468055	form_field46
45	1.680695	form_field47
46	1.137599	form_field48
47	6.904260	form_field49
48	1.069165	form_field50

In [18]:
```
#Scaling/normalizing the dataset
dd = StandardScaler().fit_transform(dd)
```

In [19]:
```
#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[19]:

	VIF Factor	features
0	1.763192	form_field1
1	1.248070	form_field2
2	3.378051	form_field3
3	2.029877	form_field4
4	3.848713	form_field5
5	1.575456	form_field6
6	6.211788	form_field7
7	2.446061	form_field8
8	3.458810	form_field9
9	8.274333	form_field10
10	1.217790	form_field11
11	1.779040	form_field12
12	1.297608	form_field13
13	1.000763	form_field14
14	1.090517	form_field15
15	8.077893	form_field16
16	99.252238	form_field17
17	94.776038	form_field18
18	47.060083	form_field19
19	44.782039	form_field20
20	2.762031	form_field21
21	3.405916	form_field22
22	2.581365	form_field23
23	1.067205	form_field24
24	3.551334	form_field25
25	7.210898	form_field26
26	7.474752	form_field27
27	1.679524	form_field28
28	2.732112	form_field29
29	1.461662	form_field30
30	1.284926	form_field31

	VIF Factor	features
31	9.192488	form_field32
32	1.285040	form_field33
33	1.856347	form_field34
34	1.644612	form_field35
35	2.887235	form_field36
36	5.729078	form_field37
37	2.207827	form_field38
38	2.167523	form_field39
39	1.281218	form_field41
40	1.886116	form_field42
41	3.601538	form_field43
42	1.857326	form_field44
43	1.414757	form_field45
44	2.397923	form_field46
45	1.096934	form_field47
46	1.115998	form_field48
47	5.216444	form_field49
48	1.060399	form_field50

In [23]:
```
#visualizing


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

Out[23]:

	form_field1	form_field2	form_field3	form_field4	form_field5	form_field6	form_f
form_field1	1.000000	-0.024816	-0.053473	-0.004178	-0.017411	0.175138	0.14
form_field2	-0.024816	1.000000	0.129518	0.097953	0.042105	-0.179908	-0.09
form_field3	-0.053473	0.129518	1.000000	0.370922	0.657757	-0.137143	-0.09
form_field4	-0.004178	0.097953	0.370922	1.000000	0.092821	-0.078984	-0.04
form_field5	-0.017411	0.042105	0.657757	0.092821	1.000000	-0.056322	-0.05
form_field6	0.175138	-0.179908	-0.137143	-0.078984	-0.056322	1.000000	0.21
form_field7	0.142078	-0.095660	-0.091465	-0.048597	-0.051617	0.211323	1.00
form_field8	0.260820	-0.190538	-0.219501	-0.123129	-0.090338	0.520730	0.36
form_field9	0.247330	-0.144224	-0.202113	-0.119278	-0.094003	0.275787	0.32
form_field10	0.196717	-0.118863	-0.143422	-0.072839	-0.071611	0.258576	0.89
form_field11	-0.100458	0.064010	0.343185	0.097705	0.231720	-0.068654	-0.05

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,missir
```

In [24]:
```
data=data.replace([np.inf, -np.inf], np.nan)

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

In [25]:
```
print(data.shape)
data.head(1)
data.columns
```

```
(80000, 40)
```

Out[25]:
```
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_field31', '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_fiel
d4_46',
       'form_field1_28', 'form_field26_27', 'form_field7_10',
       'form_field25_29'],
      dtype='object')
```

In [32]:
```
#visualizing


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

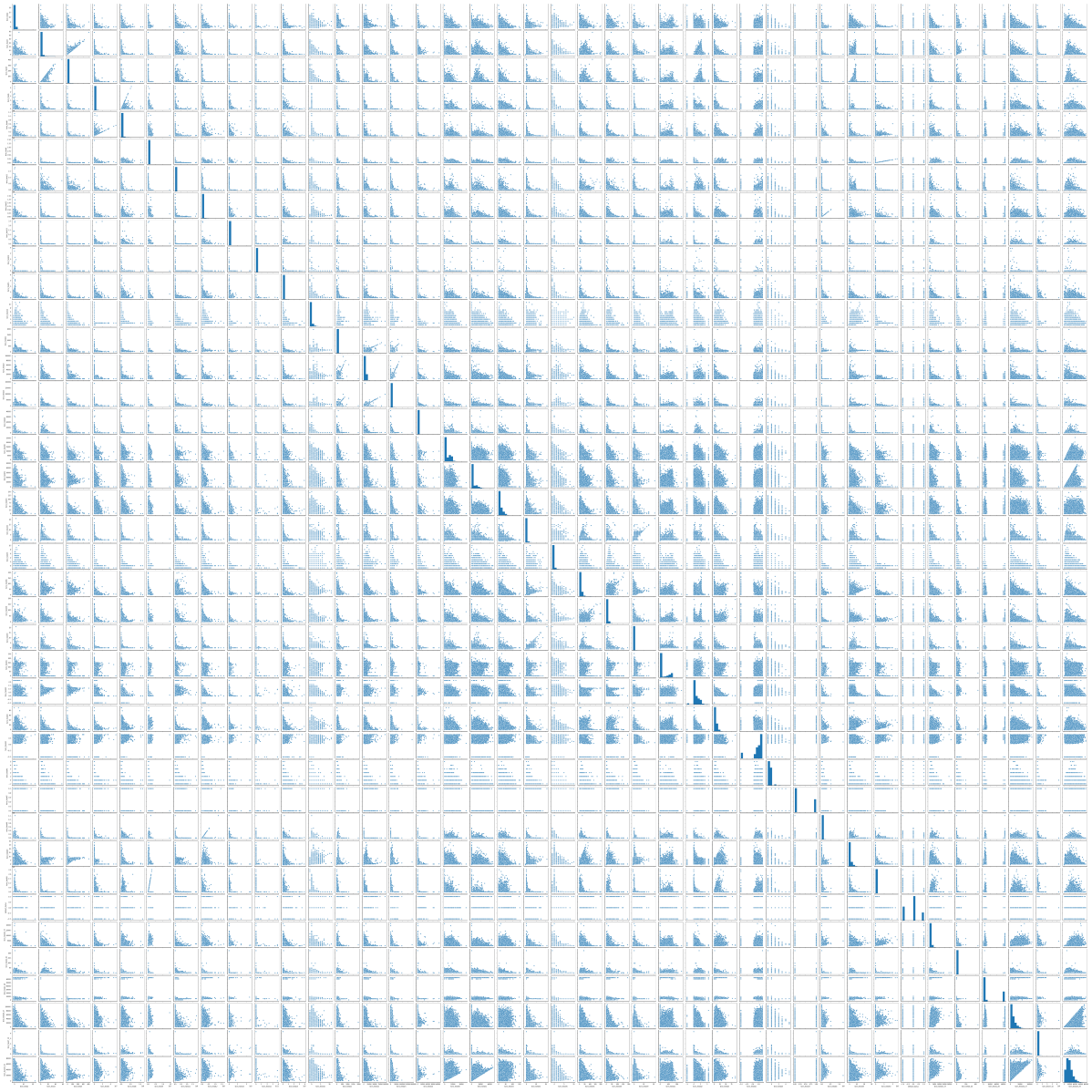
Out[32]:

	form_field2	form_field3	form_field5	form_field6	form_field8	form_field9	for
form_field2	1.000000	0.129518	0.042105	-0.179908	-0.190538	-0.144224	
form_field3	0.129518	1.000000	0.657757	-0.137143	-0.219501	-0.202113	
form_field5	0.042105	0.657757	1.000000	-0.056322	-0.090338	-0.094003	
form_field6	-0.179908	-0.137143	-0.056322	1.000000	0.520730	0.275787	
form_field8	-0.190538	-0.219501	-0.090338	0.520730	1.000000	0.589691	
form_field9	-0.144224	-0.202113	-0.094003	0.275787	0.589691	1.000000	
form_field11	0.064010	0.343185	0.231720	-0.068654	-0.112543	-0.108642	
form_field12	-0.012334	-0.166181	-0.079063	0.001588	0.249240	0.361313	
form_field13	-0.060700	-0.096396	-0.042292	0.183468	0.335028	0.261111	
form_field14	-0.005117	-0.001923	-0.002576	-0.000942	-0.001171	-0.000848	
form_field15	-0.061425	-0.089569	-0.052903	0.126409	0.218169	0.180438	
form_field16	0.104503	-0.144846	-0.096327	0.026254	0.086300	0.116069	
form_field21	0.175442	-0.053797	-0.055977	-0.197585	-0.186804	-0.060249	
form_field22	0.224258	-0.016166	-0.025852	-0.183255	-0.178043	-0.109897	
form_field23	0.236384	-0.010287	-0.020537	-0.164721	-0.153644	-0.097912	
form_field24	0.034246	0.135528	0.002979	-0.045735	-0.060306	-0.031759	
form_field30	-0.005206	0.019145	-0.037816	0.022417	0.058588	0.142066	
form_field31	0.048843	0.199213	0.241843	-0.047119	-0.072182	-0.066969	
form_field33	-0.160941	-0.101233	-0.064616	0.161045	0.214576	0.263351	
form_field34	0.084486	0.104840	0.020386	-0.083937	-0.100854	-0.051384	
form_field35	0.189518	-0.010420	-0.041483	-0.091652	0.004091	0.043682	
form_field36	0.185742	0.378104	0.238339	-0.162606	-0.134206	-0.024710	
form_field38	0.334003	0.120812	0.036335	-0.153268	-0.077239	0.031252	
form_field39	0.069993	0.040464	0.005829	0.004513	0.015903	0.034864	
form_field41	0.129898	-0.043028	-0.035655	-0.052651	-0.032373	-0.003457	
form_field42	0.128507	0.423842	0.224572	-0.209001	-0.326679	-0.326905	
form_field43	0.052726	-0.233118	-0.128282	0.038733	0.283757	0.485751	
form_field44	-0.001707	-0.108409	-0.090959	0.125539	0.165282	0.141349	
form_field45	-0.116462	-0.068392	-0.094056	0.141027	0.208063	0.284792	
form_field47	-0.111985	-0.117764	-0.052531	0.139174	0.175336	0.170319	
form_field48	-0.038087	-0.055372	-0.024134	0.031994	0.129487	0.151583	

	form_field2	form_field3	form_field5	form_field6	form_field8	form_field9	for
form_field49	0.164259	0.212196	0.477779	-0.179345	-0.171377	-0.058978	
form_field50	-0.027295	-0.038771	-0.015934	0.066216	0.135860	0.224471	
default_status	0.101508	0.082986	0.030407	-0.078214	-0.109586	-0.103519	
form_field32_37	-0.152182	-0.115110	-0.099269	0.185109	0.483046	0.713305	
form_field4_46	0.102603	0.364349	0.086057	-0.077177	-0.115901	-0.104632	
form_field1_28	-0.180924	-0.308720	-0.119174	0.319913	0.427234	0.346288	
form_field26_27	-0.188820	-0.105304	-0.095893	0.257256	0.441653	0.538406	
form_field7_10	-0.112458	-0.125791	-0.065267	0.245974	0.443349	0.453136	
form_field25_29	-0.191307	-0.067947	-0.057820	0.260556	0.389891	0.445363	

In [26]:
```
sns.pairplot(data)
```

Out[26]:
```
<seaborn.axisgrid.PairGrid at 0x7f691722e590>
```



model

In [26]:
```
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())
```

```
56000 56000
39

/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,
```

In [27]:
```
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 [35]:
```
#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/c4/1310a054d33abc318426a956e7d6df0df76a6ddfa9
```

20.4.0-py2.py3-none-any.whl (https://files.pythonhosted.org/packages/15/c4/1310a054d33abc318426a956e7d6df0df76a6ddfa920.4.0-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/d2/17/37ee6c79cafbd9bb7423b54e55ea90beec66aa7638/0.36.0.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/f37ac7f61214ed044b0df91252ab19376de5587926c/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 [43]:
```
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 [42]:
```
class ModelOptimizer:
    best_score = None
    opt = None

    def __init__(self, model, X_train, y_train, categorical_columns_indi
        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 [44]:
```
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: 230ms    remaining: 15m 20s
1:      total: 344ms    remaining: 11m 27s
2:      total: 461ms    remaining: 10m 13s
3:      total: 569ms    remaining: 9m 28s
4:      total: 672ms    remaining: 8m 56s
5:      total: 767ms    remaining: 8m 30s
6:      total: 858ms    remaining: 8m 9s
7:      total: 967ms    remaining: 8m 2s
8:      total: 1.06s    remaining: 7m 50s
9:      total: 1.16s    remaining: 7m 43s
10:     total: 1.28s    remaining: 7m 45s
11:     total: 1.38s    remaining: 7m 38s
12:     total: 1.49s    remaining: 7m 37s
13:     total: 1.6s     remaining: 7m 35s
14:     total: 1.7s     remaining: 7m 31s
15:     total: 1.82s    remaining: 7m 33s
16:     total: 1.94s    remaining: 7m 34s
17:     total: 2.04s    remaining: 7m 32s
18:     total: 2.15s    remaining: 7m 29s
19:     total: 2.26s    remaining: 7m 30s
```

In [45]:
```
# 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 [46]:
```
#calculate scores. 100% correct prediction has auc score of 1
print('Auc on test set: {:.4f}'.format(roc_auc_score(y_test, pred_proba)
```

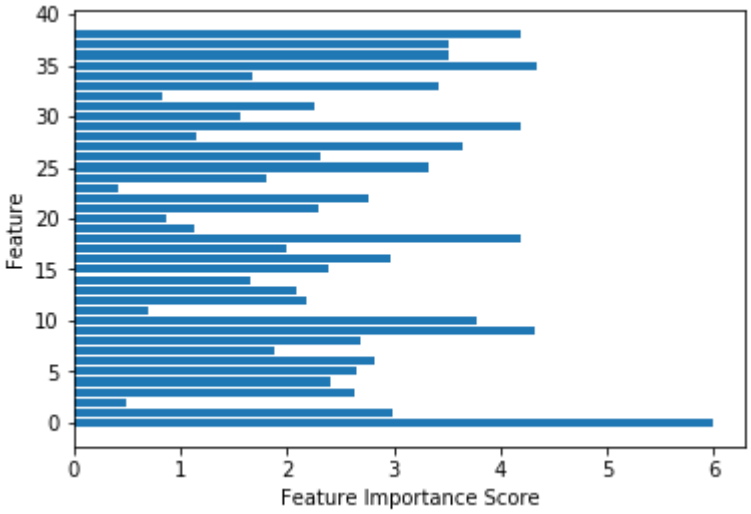
```
Auc on test set: 0.8355
```

In [ ]:
```

```

In [47]:
```
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 [66]:
```
y=pd.DataFrame(y)
y.head()
```

Out[66]:

	default_status
0	0.0
1	0.0
2	1.0
3	0.0
4	0.0

In [147]:
```
y_not_pay=y.copy()[(y["default_status"]==0.0)]
y_not_pay.rename(columns={'default_status':'y_not_pay'}, inplace=True)
y_not_pay.head()
y_not_pay.shape
```

Out[147]: (42285, 1)

In [142]:
```
y_pay=y.copy()[(y['default_status']==1.0)]
y_pay.rename(columns={'default_status':'y_pay'}, inplace=True)
y_pay.head()
y_pay.shape
```

Out[142]: (13715, 1)

In [ ]:

In [ ]:

In [43]:
```
'''for pickle'''
with open ('Mart1rfnow.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('Mart1rfnow.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('Mart1rfnow.csv', index=False)
```

In [ ]:
```

```