1 Logistic Regression Models

1.1 Importing Data

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

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import roc_curve, auc
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_sco
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.utils import resample
from imblearn.over_sampling import SMOTE

from sklearn.ensemble import BaggingClassifier, RandomForestClassifier
from sklearn.model_selection import GridSearchCV, cross_val_score
from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier
from xgboost import XGBClassifier

from sklearn import tree
import xgboost as xgb
from numpy import loadtxt
from xgboost import XGBClassifier
from xgboost import plot_tree

import gc
from tqdm import tqdm

from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.metrics import mean_squared_error, recall_score

from sklearn.model_selection import cross_val_predict
from keras import models
from keras import layers
from keras import regularizers
from keras.wrappers.scikit_learn import KerasRegressor
from keras.models import load_model
from scipy import stats
import statsmodels.api as sm
```

executed in 7.17s, finished 03:59:56 2021-04-22

In [2]:
```
column_defs = pd.read_excel('data\LCDataDictionary.xlsx',index_col='LoanStatNew')
column_defs.columns
```

executed in 47ms, finished 03:59:56 2021-04-22

Out[2]:
```
Index(['Description'], dtype='object')
```

In [3]:
```
def column_info(col_name):
    return column_defs.loc[col_name]['Description']
```
executed in 14ms, finished 03:59:56 2021-04-22

In [4]:
```
def na_check(data):
    check = np.round(data.isna().sum().sort_values(ascending=False),2)
    return check
```
executed in 14ms, finished 03:59:56 2021-04-22

In [5]:

```
def reduce_mem_usage(df, int_cast=True, obj_to_category=False, subset=None):
    """
    Iterate through all the columns of a dataframe and modify the data type to re
    :param df: dataframe to reduce (pd.DataFrame)
    :param int_cast: indicate if columns should be tried to be casted to int (boo
    :param obj_to_category: convert non-datetime related objects to category dtyp
    :param subset: subset of columns to analyse (list)
    :return: dataset with the column dtypes adjusted (pd.DataFrame)
    """
    start_mem = df.memory_usage().sum() / 1024 ** 2;
    gc.collect()
    print('Memory usage of dataframe is {:.2f} MB'.format(start_mem))

    cols = subset if subset is not None else df.columns.tolist()

    for col in tqdm(cols):
        col_type = df[col].dtype

        if col_type != object and col_type.name != 'category' and 'datetime' not
            c_min = df[col].min()
            c_max = df[col].max()

            # test if column can be converted to an integer
            treat_as_int = str(col_type)[:3] == 'int'
            if int_cast and not treat_as_int:
                treat_as_int = check_if_integer(df[col])

            if treat_as_int:
                if c_min > np.iinfo(np.int8).min and c_max < np.iinfo(np.int8).ma
                    df[col] = df[col].astype(np.int8)
                elif c_min > np.iinfo(np.uint8).min and c_max < np.iinfo(np.uint8
                    df[col] = df[col].astype(np.uint8)
                elif c_min > np.iinfo(np.int16).min and c_max < np.iinfo(np.int16
                    df[col] = df[col].astype(np.int16)
                elif c_min > np.iinfo(np.uint16).min and c_max < np.iinfo(np.uint
                    df[col] = df[col].astype(np.uint16)
                elif c_min > np.iinfo(np.int32).min and c_max < np.iinfo(np.int32
                    df[col] = df[col].astype(np.int32)
                elif c_min > np.iinfo(np.uint32).min and c_max < np.iinfo(np.uint
                    df[col] = df[col].astype(np.uint32)
                elif c_min > np.iinfo(np.int64).min and c_max < np.iinfo(np.int64
                    df[col] = df[col].astype(np.int64)
                elif c_min > np.iinfo(np.uint64).min and c_max < np.iinfo(np.uint
                    df[col] = df[col].astype(np.uint64)
            else:
                if c_min > np.finfo(np.float32).min and c_max < np.finfo(np.float
                    df[col] = df[col].astype(np.float32)
                else:
                    df[col] = df[col].astype(np.float64)
        elif 'datetime' not in col_type.name and obj_to_category:
            df[col] = df[col].astype('category')
    gc.collect()
    end_mem = df.memory_usage().sum() / 1024 ** 2
    print('Memory usage after optimization is: {:.3f} MB'.format(end_mem))
    print('Decreased by {:.1f}%'.format(100 * (start_mem - end_mem) / start_mem))
```

```
    return df
```
executed in 27ms, finished 03:59:56 2021-04-22

In [6]:
```
df = pd.read_csv('data/preprocessed.csv')
df.head()
```
executed in 9.99s, finished 03:22:00 2021-04-14

Out[6]:

	Unnamed: 0	Unnamed: 0.1	id	loan_amnt	term	int_rate	installment	grade	s
0	0	42536	10129454	12000.0	36 months	10.99%	392.8	B	
1	1	42537	10149488	4800.0	36 months	10.99%	157.1	B	
2	2	42538	10149342	27060.0	36 months	10.99%	885.5	B	
3	3	42539	10148122	12000.0	36 months	7.62%	374.0	A	
4	4	42540	10129477	14000.0	36 months	12.85%	470.8	B	

5 rows × 75 columns

In [7]:
```
df.drop(columns=['pymnt_plan','out_prncp','Unnamed: 0','Unnamed: 0.1','id'],axis=
```
executed in 398ms, finished 03:22:01 2021-04-14

In [8]:
```
df.drop(columns=["debt_settlement_flag", "hardship_flag",],axis=1,inplace=True)
```
executed in 462ms, finished 03:22:03 2021-04-14

In [9]:
```
# making average fico score and dropping the fico range high and low
df['average_fico'] = (df['fico_range_high'] + df['fico_range_low'])/2
df.drop(columns=['fico_range_high','fico_range_low'],axis=1,inplace=True)
```
executed in 1.08s, finished 03:22:05 2021-04-14

In [10]:
```
df.drop('initial_list_status',axis=1,inplace=True)
```
executed in 511ms, finished 03:22:07 2021-04-14

In [11]:
```
df.drop('title',axis=1,inplace=True)
```
executed in 429ms, finished 03:22:08 2021-04-14

In [12]:
```
df.drop(columns=['zip_code'],axis=1,inplace=True)
```
executed in 393ms, finished 03:22:10 2021-04-14

In [13]:
```
df.int_rate = df.int_rate.map(lambda x: float(x.replace('%','')))
```
executed in 892ms, finished 03:22:12 2021-04-14

In [14]:
```
regions = pd.read_excel('data/state_regions.xlsx')
```
executed in 28ms, finished 03:22:13 2021-04-14

In [15]:
```python
df['region'] = df.addr_state.apply(lambda x: regions.loc[regions['State Code'] ==
```
executed in 7m 8s, finished 03:29:22 2021-04-14

In [16]:
```python
df.drop(columns = ['addr_state'],axis=1,inplace=True)
```
executed in 1.08s, finished 03:29:24 2021-04-14

In [17]:
```python
reduce_mem_usage(df,int_cast=False)
```
executed in 7.78s, finished 03:29:33 2021-04-14

```
  0%|          | 0/64 [00:00<?, ?it/s]

Memory usage of dataframe is 848.11 MB

100%|██████████| 64/64 [00:06<00:00,  9.26it/s]

Memory usage after optimization is: 511.850 MB
Decreased by 39.6%
```

Out[17]:

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_length	hom
0	12000.0	36 months	10.99	392.799988	B	B2	4	
1	4800.0	36 months	10.99	157.100006	B	B2	2	
2	27060.0	36 months	10.99	885.500000	B	B2	10	
3	12000.0	36 months	7.62	374.000000	A	A3	3	
4	14000.0	36 months	12.85	470.799988	B	B4	4	
...	...	...	...	...	...	...	...	
1736932	24000.0	60 months	23.99	690.500000	E	E2	1	
1736933	10000.0	36 months	7.99	313.200012	A	A5	10	
1736934	10050.0	36 months	16.99	358.200012	D	D1	8	
1736935	6000.0	36 months	11.44	197.800003	B	B4	5	
1736936	30000.0	60 months	25.49	889.000000	E	E4	4	

1736937 rows × 64 columns

In [19]:
```
post_app_drops = ['total_bal_ex_mort', 'pct_tl_nvr_dlq','num_tl_op_past_12m','nur
,'num_tl_120dpd_2m','num_bc_sats','num_accts_ever_120_pd','mths_since_recent_bc',
,'chargeoff_within_12_mths','bc_util','avg_cur_bal','tot_cur_bal','tot_coll_amt',
,'last_pymnt_amnt','recoveries','total_rec_late_fee'
,'total_rec_int','total_rec_prncp','total_pymnt','revol_util','revol_bal']
```
executed in 14ms, finished 21:49:47 2021-04-12

In [18]:
```
df.drop(columns=['issue_d','earliest_cr_line','last_pymnt_d','last_credit_pull_d'
```
executed in 484ms, finished 03:29:35 2021-04-14

In [19]:
```
df.term = df.term.map(lambda x: np.int8(x.replace(' months','')))
```
executed in 1.94s, finished 03:29:37 2021-04-14

In [20]:
```
df.revol_util = df.revol_util.map(lambda x: float(x.replace('%','')))
```
executed in 837ms, finished 03:29:39 2021-04-14

In [ ]:
```
## from model testing these features are not relevant and total pymnt,prncp, and
df.drop(columns=['total_rec_int','total_rec_prncp','total_pymnt',
                  'tot_coll_amt','num_il_tl','delinq_amnt','tax_liens','last_pymnt
```

In [47]:
```
df.to_csv('data/full_clean_pre_z')
```
executed in 53.2s, finished 03:49:40 2021-04-14

In [ ]:

Start Here loading DF

In [6]:
```
df = pd.read_csv('data/full_clean_pre_z')
df.head()
```
executed in 6.52s, finished 04:00:06 2021-04-22

Out[6]:

	Unnamed: 0	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_length
0	0	12000.0	36	10.99	392.8	B	B2	
1	1	4800.0	36	10.99	157.1	B	B2	
2	2	27060.0	36	10.99	885.5	B	B2	
3	3	12000.0	36	7.62	374.0	A	A3	
4	4	14000.0	36	12.85	470.8	B	B4	

5 rows × 51 columns

In [7]:
```
df.drop('Unnamed: 0',axis=1,inplace=True)
```
executed in 266ms, finished 04:00:07 2021-04-22

In [8]:

```python
reduce_mem_usage(df,int_cast=False)
```

executed in 6.25s, finished 04:00:14 2021-04-22

```
  0%|          | 0/50 [00:00<?, ?it/s]

Memory usage of dataframe is 662.59 MB

100%|██████████| 50/50 [00:05<00:00,  9.16it/s]

Memory usage after optimization is: 374.363 MB
Decreased by 43.5%
```

Out[8]:

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_length	hom
0	12000.0	36	10.99	392.799988	B	B2	4	
1	4800.0	36	10.99	157.100006	B	B2	2	
2	27060.0	36	10.99	885.500000	B	B2	10	
3	12000.0	36	7.62	374.000000	A	A3	3	
4	14000.0	36	12.85	470.799988	B	B4	4	
...	...	...	...	...	...	...	...	
1736932	24000.0	60	23.99	690.500000	E	E2	1	
1736933	10000.0	36	7.99	313.200012	A	A5	10	
1736934	10050.0	36	16.99	358.200012	D	D1	8	
1736935	6000.0	36	11.44	197.800003	B	B4	5	
1736936	30000.0	60	25.49	889.000000	E	E4	4	

1736937 rows × 50 columns

In [9]:
```
df.info()
```
executed in 14ms, finished 04:00:14 2021-04-22

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1736937 entries, 0 to 1736936
Data columns (total 50 columns):
 #   Column                       Dtype
---  ------                       -----
 0   loan_amnt                    float32
 1   term                         int8
 2   int_rate                     float32
 3   installment                  float32
 4   grade                        object
 5   sub_grade                    object
 6   emp_length                   int8
 7   home_ownership               object
 8   annual_inc                   float32
 9   verification_status          object
 10  loan_status                  object
 11  purpose                      object
 12  dti                          float32
 13  delinq_2yrs                  float32
```

In [10]:
```
objects = list(df.loc[:,df.dtypes == 'object'].columns)
```
executed in 273ms, finished 04:00:17 2021-04-22

In [11]:
```
objects
```
executed in 12ms, finished 04:00:17 2021-04-22

Out[11]:
```
['grade',
 'sub_grade',
 'home_ownership',
 'verification_status',
 'loan_status',
 'purpose',
 'application_type',
 'region']
```

In [12]:
```
categorical = ['sub_grade',
 'grade',
 'home_ownership',
 'verification_status',
 'purpose',
 'application_type',
 'region']


cat_drop = ['sub_grade',
 'grade',
 'home_ownership',
 'verification_status',
 'purpose',
 'application_type',
 'region','loan_status']
```
executed in 13ms, finished 04:00:18 2021-04-22

In [13]:
```
onehot = pd.get_dummies(df[categorical],drop_first=True)
```
executed in 1.14s, finished 04:00:21 2021-04-22

In [14]:
```
cont_columns = df.drop(columns=cat_drop).columns
```
executed in 156ms, finished 04:00:22 2021-04-22

In [15]:
```
df_cont_z = df[(np.abs(stats.zscore(df[cont_columns]))<4).all(axis=1)]
```
executed in 1.31s, finished 04:00:24 2021-04-22

In [16]:
```
z_score_df = df.loc[df_cont_z.index]
```
executed in 330ms, finished 04:00:25 2021-04-22

In [17]:
```
onehot_z = pd.get_dummies(z_score_df[categorical],drop_first=True)
```
executed in 1.11s, finished 04:00:28 2021-04-22

In [18]:
```
cont_z = z_score_df.drop(columns=cat_drop)
```
executed in 111ms, finished 04:00:30 2021-04-22

In [19]:
```
xz= pd.concat([cont_z,onehot_z],axis=1)
yz= z_score_df['loan_status'].map(lambda x: 1 if x== "Charged Off" else 0)
```
executed in 595ms, finished 04:00:32 2021-04-22

In [20]:
```
x_train_z, x_test_z, y_train_z, y_test_z = train_test_split(xz, yz, test_size=0.3
```
executed in 951ms, finished 04:00:35 2021-04-22

In [21]:
```
cat_index = np.linspace(50,113,num=(113-50)).astype(np.int)
cat_index
```
executed in 16ms, finished 04:00:37 2021-04-22

Out[21]:
```
array([ 50,  51,  52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,
        63,  64,  65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,
        76,  77,  78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,
        89,  90,  91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101,
       102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 113])
```

2 Using Smote to Oversample Minority Class (do not run)

In [42]:
```
from imblearn.over_sampling import SMOTENC
```
executed in 15ms, finished 21:53:31 2021-04-12

In [43]:
```
smote= SMOTENC(categorical_features=cat_index,k_neighbors=4)
x_smote_z,y_smote_z = smote.fit_sample(x_train_z,y_train_z)
```
executed in 2h 40m 37s, finished 00:35:57 2021-04-13

In [44]:
```
x_smote_z.to_csv('data/x_smote_z')
y_smote_z.to_csv('data/y_smote_z')
```
executed in 1m 13.8s, finished 00:37:24 2021-04-13

In [ ]:

In [22]:
```python
x_smote_z = pd.read_csv('data/x_smote_z')
y_smote_z = pd.read_csv('data/y_smote_z')
```
executed in 21.2s, finished 04:01:01 2021-04-22

In [23]:
```python
# as we dropped columns after testing to our original df need to drop from smote
# at end will need to re run smote
x_smote_z.drop(columns=['total_rec_int','total_rec_prncp','total_pymnt',
                 'tot_coll_amt','num_il_tl','delinq_amnt','tax_liens','last_pymnt
```
executed in 505ms, finished 04:01:04 2021-04-22

In [24]:
```python
reduce_mem_usage(x_smote_z,int_cast=False)
```
executed in 16.1s, finished 04:01:23 2021-04-22

```
  0%|          | 0/107 [00:00<?, ?it/s]

Memory usage of dataframe is 1327.29 MB

100%|██████████| 107/107 [00:15<00:00,  7.00it/s]

Memory usage after optimization is: 356.632 MB
Decreased by 73.1%
```

Out[24]:

	Unnamed: 0	loan_amnt	term	int_rate	installment	emp_length	annual_inc
0	0	7750.000000	36	10.990000	253.800003	3	29000.000000
1	1	10000.000000	36	22.910000	386.799988	1	72000.000000
2	2	10000.000000	36	12.990000	337.000000	6	40000.000000
3	3	5500.000000	36	16.549999	194.899994	4	54224.000000
4	4	25000.000000	36	14.520000	861.000000	5	75000.000000
...	...	...	...	...	...	...	...
1625891	1625891	24366.757812	51	17.206345	707.872986	4	117479.695312
1625892	1625892	19482.703125	50	15.419235	542.377869	5	51204.097656
1625893	1625893	13179.145508	36	14.562667	461.734833	6	55000.000000
1625894	1625894	17840.857422	45	21.473141	597.083801	8	54589.417969
1625895	1625895	20000.000000	60	22.391314	557.422424	9	62470.488281

1625896 rows × 107 columns

In [25]:
```python
x_smote_z.drop('Unnamed: 0',axis=1,inplace=True)
```
executed in 564ms, finished 04:01:26 2021-04-22

In [26]:
```
y_smote_z.drop('Unnamed: 0',axis=1,inplace=True)
```
executed in 15ms, finished 04:01:28 2021-04-22

In [27]:
```
reduce_mem_usage(y_smote_z,int_cast=False)
```
executed in 251ms, finished 04:01:30 2021-04-22

```
100%|██████████| 1/1 [00:00<00:00, 62.50it/s]

Memory usage of dataframe is 12.40 MB
Memory usage after optimization is: 1.551 MB
Decreased by 87.5%
```

Out[27]:

	loan_status
0	0
1	1
2	0
3	1
4	1
...	...
1625891	1
1625892	1
1625893	1
1625894	1
1625895	1

1625896 rows × 1 columns

In [28]:
```
scaler = MinMaxScaler()
```
executed in 16ms, finished 04:01:33 2021-04-22

In [29]:
```
x_smote_cont_z =  pd.DataFrame(scaler.fit_transform(x_smote_z[cont_columns]),colu
x_smote_scaled_z = pd.concat([x_smote_cont_z,x_smote_z[onehot.columns]],axis=1)
```
executed in 889ms, finished 04:01:36 2021-04-22

In [30]:
```
x_test_z= x_test_z.reset_index().drop('index',axis=1)
x_test_cont_scaled_z = pd.DataFrame(scaler.transform(x_test_z[cont_columns]),colu
x_test_scaled_z = pd.concat([x_test_cont_scaled_z,x_test_z[onehot.columns]],axis=
```
executed in 284ms, finished 04:01:39 2021-04-22

In [31]:
```
x_train_final_z, x_val_z, y_train_final_z, y_val_z = train_test_split(x_smote_sca
```
executed in 964ms, finished 04:01:43 2021-04-22

In [32]:
```
x_train_z = x_train_z.reset_index().drop('index',axis=1)
x_cont_z =  pd.DataFrame(scaler.fit_transform(x_train_z[cont_columns]),columns=co
x_train_scaled_z = pd.concat([x_cont_z,x_train_z[onehot.columns]],axis=1)
```
executed in 748ms, finished 04:01:46 2021-04-22

In [33]:
```
y_train_z = y_train_z.reset_index().drop('index',axis=1)
```
executed in 47ms, finished 04:01:49 2021-04-22

2.1 Logistic Regression Models

In [34]:
```
z_score_df[z_score_df.tot_cur_bal ==0]['loan_status'].value_counts(normalize=True
```
executed in 35ms, finished 04:01:56 2021-04-22

Out[34]:
```
Fully Paid     0.743649
Charged Off    0.256351
Name: loan_status, dtype: float64
```

In [35]:
```
r = ['loan_amnt',
'term',
'int_rate',
'annual_inc',
'emp_length',
'dti',
'total_acc',
 'tot_cur_bal',
  'pct_tl_nvr_dlq',
 'total_rec_late_fee',
 'mort_acc',
 'mths_since_recent_inq',
 'num_tl_90g_dpd_24m',
 'pub_rec_bankruptcies',
 'total_bc_limit',
 'total_il_high_credit_limit',
 'average_fico',
'bc_util',
     'num_tl_120dpd_2m',
 'num_tl_30dpd',
 'num_tl_90g_dpd_24m',
     'tot_coll_amt',
     'grade_B',
 'grade_C',
 'grade_D',
 'grade_E',
 'grade_F',
 'grade_G',
 'home_ownership_MORTGAGE',
 'home_ownership_OWN',
 'home_ownership_RENT',
 'verification_status_Source Verified',
 'verification_status_Verified',
 'purpose_credit_card',
 'purpose_debt_consolidation',
 'purpose_home_improvement',
 'purpose_house',
 'purpose_major_purchase',
 'purpose_medical',
 'purpose_moving',
 'purpose_other',
 'purpose_renewable_energy',
 'purpose_small_business',
 'purpose_vacation',
 'purpose_wedding',
 'application_type_Joint App',
 'region_Northeast',
 'region_South',
 'region_West']
```

executed in 8ms, finished 04:02:05 2021-04-22

In [ ]:
```
for_sure_out = ['pub_rec','tot_coll_amt','num_il_tl','delinq_amnt','tax_liens', '
 'purpose_moving','region_Northeast']
```

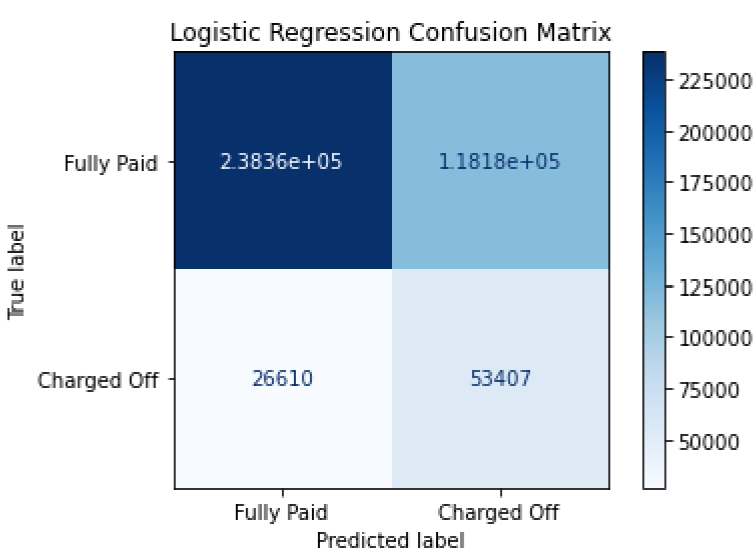
In [36]:
```
balanced_og=['loan_amnt','term','int_rate','installment','emp_length','annual_inc
 'revol_bal','average_fico','total_acc','total_rec_late_fee','tot_cur_bal','bc_ut
 'mort_acc','mths_since_recent_bc','mths_since_recent_inq',
 'num_accts_ever_120_pd','num_bc_sats','num_bc_tl','num_op_rev_tl','num_tl_90g_dp
'num_tl_op_past_12m','pct_tl_nvr_dlq','pub_rec_bankruptcies','total_bal_ex_mort',
'grade_B','grade_C','grade_D','grade_E','grade_F','grade_G','home_ownership_MORTG
'verification_status_Source Verified','verification_status_Verified','purpose_cre
'purpose_home_improvement','purpose_house','purpose_major_purchase','purpose_othe
'purpose_small_business','purpose_vacation','purpose_wedding','application_type_J
```

executed in 15ms, finished 04:02:08 2021-04-22

2.1.1 Logistic Regression unsmoted without recoveries

In [37]:
```
#best log reg using balanced on original data using test features 4std z score dr
logreg = LogisticRegression(C=1e7,fit_intercept=True,solver='lbfgs',penalty='l2',
log_model = logreg.fit(x_train_scaled_z[balanced_og], y_train_z)
y_hat_log = logreg.predict(x_test_scaled_z[balanced_og])
y_score_log = log_model.decision_function(x_test_scaled_z[balanced_og])
fpr,tpr,thresholds = roc_curve(y_test_z,y_score_log)

print('AUC: {}'.format(auc(fpr, tpr)))
cf = confusion_matrix(y_test_z,y_hat_log)

plot_confusion_matrix(log_model,x_test_scaled_z[balanced_og],y_test_z,cmap=plt.cm
                      display_labels=["Fully Paid", "Charged Off"],
                       values_format=".5g")
plt.title("Logistic Regression Confusion Matrix")

print(confusion_matrix(y_test_z, y_hat_log))
print(classification_report(y_test_z, y_hat_log))
```
executed in 32.0s, finished 04:02:54 2021-04-22

```
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\sklearn\utils\validat
ion.py:72: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  return f(**kwargs)
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worker
s.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   28.0s finished

AUC: 0.7325226073180483
[[238365 118181]
 [ 26610  53407]]
              precision    recall  f1-score   support

           0       0.90      0.67      0.77    356546
           1       0.31      0.67      0.42     80017

    accuracy                           0.67    436563
   macro avg       0.61      0.67      0.60    436563
weighted avg       0.79      0.67      0.70    436563
```



In [38]:
```
pd.DataFrame(index =x_train_scaled_z[balanced_og].columns,data=log_model.coef_.fl
```
executed in 101ms, finished 04:04:24 2021-04-22

Out[38]:

	0
loan_amnt	-0.358589
term	0.682327
int_rate	1.357631
installment	1.091914
emp_length	-0.061001
annual_inc	-0.428104
dti	1.613735
delinq_2yrs	0.139126
inq_last_6mths	0.188970
revol_bal	-0.875154
average_fico	-0.628713
total_acc	-0.642437
total_rec_late_fee	4.061917
tot_cur_bal	-0.411654
bc_util	-0.022456
mo_sin_rcnt_tl	-0.050420
mort_acc	-0.249371
mths_since_recent_bc	-0.444761
mths_since_recent_inq	-0.143758
num_accts_ever_120_pd	0.156580
num_bc_sats	0.630037
num_bc_tl	-0.167006
num_op_rev_tl	0.413646
num_tl_90g_dpd_24m	0.014719
num_tl_op_past_12m	0.459333
pct_tl_nvr_dlq	0.299752
pub_rec_bankruptcies	0.061015
total_bal_ex_mort	1.105648
total_bc_limit	-0.793603
total_il_high_credit_limit	-1.273516
grade_B	0.262358

	0
grade_C	0.420945
grade_D	0.417083
grade_E	0.345584
grade_F	0.223280
grade_G	0.116112
home_ownership_MORTGAGE	-0.322275
home_ownership_OWN	-0.201177
home_ownership_RENT	-0.077707
verification_status_Source Verified	0.049142
verification_status_Verified	0.035676
purpose_credit_card	-0.102147
purpose_debt_consolidation	-0.088981
purpose_home_improvement	-0.002114
purpose_house	-0.028096
purpose_major_purchase	-0.039484
purpose_other	-0.013780
purpose_renewable_energy	-0.115257
purpose_small_business	0.372360
purpose_vacation	0.030356
purpose_wedding	-0.534285
application_type_Joint App	0.058284
region_South	0.027277
region_West	-0.081059

In [39]:
```
logit_model=sm.Logit(y_train_z,x_train_scaled_z[balanced_og])
result=logit_model.fit(method='bfgs',maxiter=750)
print(result.summary())
```
executed in 4m 58s, finished 04:09:31 2021-04-22

```
Optimization terminated successfully.
         Current function value: 0.424408
         Iterations: 309
         Function evaluations: 310
         Gradient evaluations: 310
                           Logit Regression Results
==============================================================================
Dep. Variable:            loan_status   No. Observations:              1018645
Model:                          Logit   Df Residuals:                  1018591
Method:                           MLE   Df Model:                           53
Date:                Thu, 22 Apr 2021   Pseudo R-squ.:                  0.1087
Time:                        04:09:31   Log-Likelihood:            -4.3232e+05
converged:                       True   LL-Null:                   -4.8503e+05
Covariance Type:            nonrobust   LLR p-value:                     0.000
===============================================================================
========================
                                  coef    std err          z      P>|z|
[0.025      0.975]
-------------------------------------------------------------------------------
------------------------
loan_amnt                      -0.3736      0.089     -4.200      0.000
-0.548      -0.199
term                            0.6956      0.015     46.802      0.000
0.666       0.725
int_rate                        1.1774      0.046     25.769      0.000
1.088       1.267
installment                     1.0529      0.107      9.817      0.000
0.843       1.263
emp_length                     -0.0620      0.007     -8.662      0.000
-0.076      -0.048
annual_inc                     -0.5654      0.063     -8.949      0.000
-0.689      -0.442
dti                             1.5180      0.032     47.414      0.000
1.455       1.581
delinq_2yrs                     0.0713      0.019      3.838      0.000
0.035       0.108
inq_last_6mths                  0.1511      0.015      9.960      0.000
0.121       0.181
revol_bal                      -0.8498      0.052    -16.326      0.000
-0.952      -0.748
average_fico                   -0.6585      0.023    -28.369      0.000
-0.704      -0.613
total_acc                      -0.6092      0.033    -18.737      0.000
-0.673      -0.545
total_rec_late_fee              3.5151      0.025    142.867      0.000
3.467       3.563
tot_cur_bal                    -0.4237      0.029    -14.795      0.000
-0.480      -0.368
bc_util                        -0.0713      0.024     -2.985      0.003
-0.118      -0.024
mo_sin_rcnt_tl                 -0.1019      0.024     -4.263      0.000
-0.149      -0.055
```

```
mort_acc                                 -0.2721      0.020    -13.446      0.000
-0.312      -0.232
mths_since_recent_bc                     -0.4713      0.021    -22.069      0.000
-0.513      -0.429
mths_since_recent_inq                    -0.1679      0.014    -11.673      0.000
-0.196      -0.140
num_accts_ever_120_pd                     0.0910      0.020      4.586      0.000
0.052       0.130
num_bc_sats                               0.6179      0.034     18.302      0.000
0.552       0.684
num_bc_tl                                -0.2429      0.031     -7.920      0.000
-0.303      -0.183
num_op_rev_tl                             0.3984      0.030     13.249      0.000
0.339       0.457
num_tl_90g_dpd_24m                        0.0507      0.028      1.808      0.071
-0.004       0.106
num_tl_op_past_12m                        0.4101      0.019     21.256      0.000
0.372       0.448
pct_tl_nvr_dlq                            0.1721      0.020      8.500      0.000
0.132       0.212
pub_rec_bankruptcies                      0.0587      0.008      6.995      0.000
0.042       0.075
total_bal_ex_mort                         1.0920      0.074     14.707      0.000
0.946       1.238
total_bc_limit                           -0.7929      0.034    -22.992      0.000
-0.860      -0.725
total_il_high_credit_limit               -1.2391      0.067    -18.440      0.000
-1.371      -1.107
grade_B                                   0.2443      0.013     18.726      0.000
0.219       0.270
grade_C                                   0.4207      0.017     25.182      0.000
0.388       0.453
grade_D                                   0.4364      0.023     19.348      0.000
0.392       0.481
grade_E                                   0.3872      0.029     13.548      0.000
0.331       0.443
grade_F                                   0.2887      0.037      7.884      0.000
0.217       0.360
grade_G                                   0.1983      0.048      4.153      0.000
0.105       0.292
home_ownership_MORTGAGE                  -2.6434      0.030    -88.138      0.000
-2.702      -2.585
home_ownership_OWN                       -2.5357      0.030    -83.206      0.000
-2.595      -2.476
home_ownership_RENT                      -2.4039      0.030    -81.391      0.000
-2.462      -2.346
verification_status_Source Verified       0.0348      0.007      5.056      0.000
0.021       0.048
verification_status_Verified              0.0268      0.008      3.558      0.000
0.012       0.042
purpose_credit_card                      -0.1892      0.017    -11.181      0.000
-0.222      -0.156
purpose_debt_consolidation               -0.1723      0.016    -10.687      0.000
-0.204      -0.141
purpose_home_improvement                 -0.0968      0.020     -4.957      0.000
-0.135      -0.059
purpose_house                            -0.1444      0.039     -3.725      0.000
```

```
-0.220      -0.068
purpose_major_purchase                     -0.1229      0.025     -4.976      0.000
-0.171      -0.074
purpose_other                              -0.1050      0.019     -5.538      0.000
-0.142      -0.068
purpose_renewable_energy                   -0.2122      0.110     -1.925      0.054
-0.428       0.004
purpose_small_business                      0.2552      0.029      8.650      0.000
0.197        0.313
purpose_vacation                           -0.0637      0.036     -1.776      0.076
-0.134       0.007
purpose_wedding                            -0.6412      0.126     -5.084      0.000
-0.888      -0.394
application_type_Joint App                  0.0306      0.015      2.075      0.038
0.002        0.059
region_South                                0.0210      0.006      3.307      0.001
0.009        0.034
region_West                                -0.0868      0.007    -12.327      0.000
-0.101      -0.073
==============================================================================
========================
```

2.1.2 Logistic Regression with Recoveries Unsmoted

In [42]:

```python
balanced_og_recoveries =['recoveries',
'loan_amnt','term','int_rate','installment','emp_length','annual_inc','dti','deli
 'revol_bal','average_fico','total_acc','total_rec_late_fee','tot_cur_bal','bc_ut
 'mort_acc','mths_since_recent_bc','mths_since_recent_inq',
'num_bc_sats','num_bc_tl','num_op_rev_tl','num_tl_90g_dpd_24m',
'num_tl_op_past_12m','pct_tl_nvr_dlq','pub_rec_bankruptcies','total_bal_ex_mort',
'grade_B','grade_C','grade_D','grade_E','grade_F','grade_G','home_ownership_MORTG
'verification_status_Source Verified','verification_status_Verified','purpose_cre
'purpose_home_improvement','purpose_house','purpose_major_purchase','purpose_othe
'purpose_vacation','purpose_wedding','application_type_Joint App', 'region_South'
```

executed in 12ms, finished 04:12:38 2021-04-22

In [43]:

```
#best log reg using balanced on original data using test features 4std z score dr
logreg = LogisticRegression(C=1e7,fit_intercept=False,solver='lbfgs',penalty='l2'
                            class_weight='balanced')

log_model = logreg.fit(x_train_scaled_z[balanced_og_recoveries], y_train_z)
y_hat_log = logreg.predict(x_test_scaled_z[balanced_og_recoveries])
y_score_log = log_model.decision_function(x_test_scaled_z[balanced_og_recoveries]
fpr,tpr,thresholds = roc_curve(y_test_z,y_score_log)

print('AUC: {}'.format(auc(fpr, tpr)))
cf = confusion_matrix(y_test_z,y_hat_log)

plot_confusion_matrix(log_model,x_test_scaled_z[balanced_og_recoveries],y_test_z,
                      display_labels=["Fully Paid", "Charged Off"],
                       values_format=".5g")
plt.title("Logistic Regression Confusion Matrix")

print(confusion_matrix(y_test_z, y_hat_log))
print(classification_report(y_test_z, y_hat_log))
```

executed in 28.3s, finished 04:13:09 2021-04-22

```
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\sklearn\utils\valid
ation.py:72: DataConversionWarning: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of y to (n_samples, ), for example
using ravel().
  return f(**kwargs)
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\sklearn\linear_mode
l\_logistic.py:762: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
```

In [44]:
```python
# Feature Coefficients for sklearn log reg model
pd.DataFrame(index =x_train_scaled_z[balanced_og_recoveries].columns,data=log_moc
```
executed in 104ms, finished 04:13:28 2021-04-22

Out[44]:

	0
recoveries	656.337877
loan_amnt	0.747454
term	0.649438
int_rate	2.783838
installment	-0.199924
emp_length	-0.061298
annual_inc	-1.545751
dti	1.537363
delinq_2yrs	-0.155089
revol_bal	-0.209696

In [45]:
```python
# Our smoted model has different pvalues for various features, so we will drop fe
#confidence level
result.pvalues.sort_values(ascending=False).head(10)
```
executed in 16ms, finished 04:13:31 2021-04-22

Out[45]:
```
purpose_vacation                0.075808
num_tl_90g_dpd_24m              0.070646
purpose_renewable_energy        0.054241
application_type_Joint App      0.037993
bc_util                         0.002840
region_South                    0.000944
verification_status_Verified    0.000374
purpose_house                   0.000196
delinq_2yrs                     0.000124
grade_G                         0.000033
dtype: float64
```

In [46]:
```
logit_model=sm.Logit(y_train_z,x_train_scaled_z[balanced_og_recoveries])
result=logit_model.fit(method='lbfgs',maxiter=750)
print(result.summary())
```
executed in 11m 46s, finished 04:25:21 2021-04-22

```
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\statsmodels\base\mo
del.py:566: ConvergenceWarning: Maximum Likelihood optimization failed to con
verge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

2.1.3 Logistic Regression Model for Smoted values with recoveries

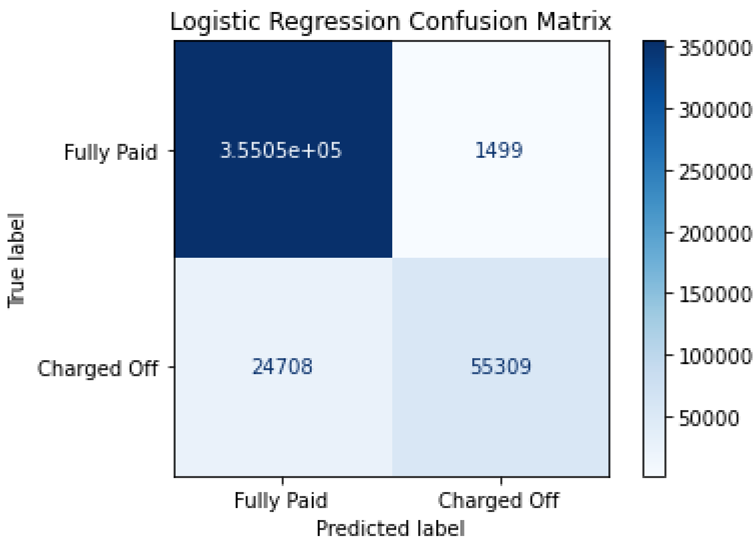
In [47]:
```
recover = ['recoveries',
'loan_amnt','term','int_rate','installment','emp_length','annual_inc','dti','deli
'pub_rec', 'revol_bal','total_acc','total_rec_late_fee','tot_cur_bal','bc_util',
 'mths_since_recent_bc', 'num_accts_ever_120_pd',
 'num_bc_sats', 'num_bc_tl',
 'num_op_rev_tl',
 'num_tl_op_past_12m', 'pct_tl_nvr_dlq',
 'total_bal_ex_mort', 'total_bc_limit',
 'total_il_high_credit_limit','average_fico',
 'sub_grade_A2',
 'sub_grade_A3','sub_grade_A4','sub_grade_A5','sub_grade_B1','sub_grade_B2','sub_
 'sub_grade_C1','sub_grade_C2','sub_grade_C3','sub_grade_C4','sub_grade_C5','sub_
 'sub_grade_D3','sub_grade_D4','sub_grade_D5','sub_grade_E1'
,'sub_grade_E2','sub_grade_E3','sub_grade_E4','sub_grade_E5','sub_grade_F1','sub_
 'sub_grade_F5','sub_grade_G1','sub_grade_G2','sub_grade_G3','sub_grade_G4','sub_
 'home_ownership_MORTGAGE','home_ownership_OWN','home_ownership_RENT',
'verification_status_Source Verified','verification_status_Verified',
 'purpose_credit_card','purpose_debt_consolidation','purpose_home_improvement','p
'purpose_other','purpose_renewable_energy','purpose_small_business','purpose_vaca
'purpose_moving',
'application_type_Joint App',
 'region_South','region_West','region_Northeast']
```
executed in 49ms, finished 04:26:32 2021-04-22

In [48]:

```python
#log reg model including recoveries
logreg = LogisticRegression(C=10,fit_intercept=False,solver='lbfgs',penalty='l2',
log_model = logreg.fit(x_train_final_z[recover], y_train_final_z.loan_status)
y_hat_log = logreg.predict(x_test_scaled_z[recover])
y_score_log = log_model.decision_function(x_test_scaled_z[recover])
fpr,tpr,thresholds = roc_curve(y_test_z,y_score_log)

print('AUC: {}'.format(auc(fpr, tpr)))
cf = confusion_matrix(y_test_z,y_hat_log)

plot_confusion_matrix(log_model,x_test_scaled_z[recover],y_test_z,cmap=plt.cm.Blu
                      display_labels=["Fully Paid", "Charged Off"],
                       values_format=".5g")
plt.title("Logistic Regression Confusion Matrix")

print(confusion_matrix(y_test_z, y_hat_log))
print(classification_report(y_test_z, y_hat_log))
```

executed in 1m 12.7s, finished 04:27:48 2021-04-22

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worker
s.
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\sklearn\linear_model
\_logistic.py:762: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:  1.1min finished

AUC: 0.9009224782285918
[[355047   1499]
 [ 24708  55309]]
              precision    recall  f1-score   support

           0       0.93      1.00      0.96    356546
           1       0.97      0.69      0.81     80017

    accuracy                           0.94    436563
   macro avg       0.95      0.84      0.89    436563
weighted avg       0.94      0.94      0.94    436563
```



In [49]:
```python
pd.options.display.max_rows = 85
```
executed in 13ms, finished 04:28:45 2021-04-22

In [50]:
```python
pd.DataFrame(index= x_train_final_z[recover].columns,data=log_model.coef_.flatten
```
executed in 154ms, finished 04:28:48 2021-04-22

Out[50]:

	0
recoveries	224.589283
loan_amnt	2.907924
term	0.547196
int_rate	8.724993
installment	-2.477596
emp_length	-0.110124
annual_inc	-1.144473
dti	2.570675
delinq_2yrs	0.210603
inq_last_6mths	0.294931

In [51]:
```
logit_model=sm.Logit(y_train_final_z,x_train_final_z[recover])
result=logit_model.fit(method='lbfgs',maxiter=250)
print(result.summary())
```

executed in 7m 32s, finished 04:36:28 2021-04-22

```
C:\Users\sergi\anaconda3\envs\learn-env\lib\site-packages\statsmodels\base\mo
del.py:566: ConvergenceWarning: Maximum Likelihood optimization failed to con
verge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

2.1.4 Logistic Regression Model Smoted values, without Recoveries

In [52]:
```
resamp =['loan_amnt','term','int_rate','installment','emp_length','annual_inc','c
'pub_rec', 'revol_bal','total_acc','total_rec_late_fee','tot_cur_bal','bc_util','
 'mths_since_recent_bc', 'num_accts_ever_120_pd',
 'num_bc_sats', 'num_bc_tl',
 'num_op_rev_tl', 'num_tl_90g_dpd_24m',
 'num_tl_op_past_12m', 'pct_tl_nvr_dlq',
 'total_bal_ex_mort', 'total_bc_limit',
 'total_il_high_credit_limit','average_fico',
 'sub_grade_A2',
 'sub_grade_A3','sub_grade_A4','sub_grade_A5','sub_grade_B1','sub_grade_B2','sub_
 'sub_grade_C1','sub_grade_C2','sub_grade_C3','sub_grade_C4','sub_grade_C5','sub_
 'sub_grade_D3','sub_grade_D4','sub_grade_D5','sub_grade_E1'
,'sub_grade_E2','sub_grade_E3','sub_grade_E4','sub_grade_E5','sub_grade_F1','sub_
 'sub_grade_F5','sub_grade_G1','sub_grade_G2','sub_grade_G3','sub_grade_G4','sub_
 'home_ownership_MORTGAGE','home_ownership_OWN','home_ownership_RENT',
'verification_status_Source Verified','verification_status_Verified',
 'purpose_credit_card','purpose_debt_consolidation','purpose_home_improvement','p
'purpose_other','purpose_renewable_energy','purpose_small_business','purpose_vaca
'purpose_moving',
'application_type_Joint App',
 'region_South','region_West','region_Northeast']
```

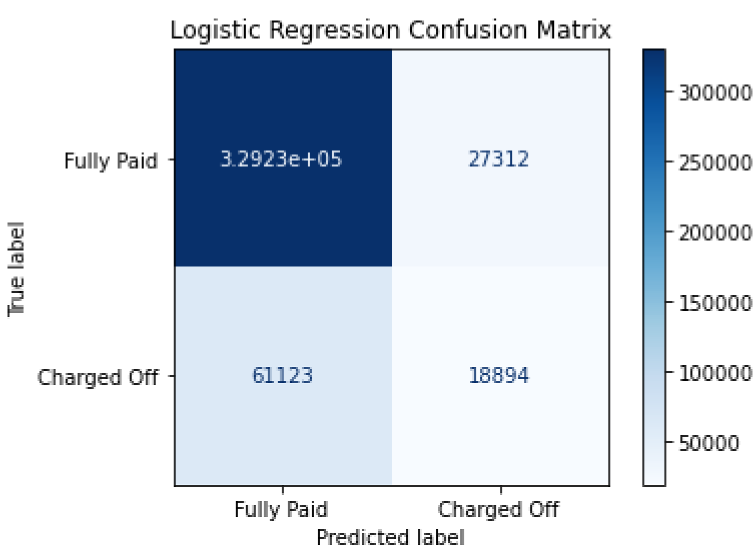
executed in 16ms, finished 04:44:48 2021-04-22

In [54]:

```
#smote log reg without recoveries
logreg = LogisticRegression(C=.75,fit_intercept=False,solver='saga',penalty='l1',
log_model = logreg.fit(x_train_final_z[resamp], y_train_final_z.loan_status)
y_hat_log = logreg.predict(x_test_scaled_z[resamp])
y_score_log = log_model.decision_function(x_test_scaled_z[resamp])
fpr,tpr,thresholds = roc_curve(y_test_z,y_score_log)

print('AUC: {}'.format(auc(fpr, tpr)))
cf = confusion_matrix(y_test_z,y_hat_log)

plot_confusion_matrix(log_model,x_test_scaled_z[resamp],y_test_z,cmap=plt.cm.Blue
                     display_labels=["Fully Paid", "Charged Off"],
                      values_format=".5g")
plt.title("Logistic Regression Confusion Matrix")

print(confusion_matrix(y_test_z, y_hat_log))
print(classification_report(y_test_z, y_hat_log))
```

executed in 31.0s, finished 04:45:36 2021-04-22

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worker
s.

convergence after 18 epochs took 28 seconds

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   28.1s finished

AUC: 0.6900494207748351
[[329234  27312]
 [ 61123  18894]]
              precision    recall  f1-score   support

           0       0.84      0.92      0.88    356546
           1       0.41      0.24      0.30     80017

    accuracy                           0.80    436563
   macro avg       0.63      0.58      0.59    436563
weighted avg       0.76      0.80      0.77    436563
```



In [262]:
```
pd.options.display.max_rows= 85
```
executed in 14ms, finished 05:46:55 2021-04-13

In [263]:
```
pd.DataFrame(index =x_train_final_z[resamp].columns,data=log_model.coef_.flatten(
```
executed in 191ms, finished 05:47:01 2021-04-13

Out[263]:

	0
loan_amnt	1.891263
term	0.535819
int_rate	6.995911
installment	-1.193287
emp_length	-0.136702
annual_inc	0.313413
dti	2.583270
delinq_2yrs	0.644501
inq_last_6mths	0.344524
revol_bal	-1.567845

```
In [55]: logit_model=sm.Logit(y_train_final_z,x_train_final_z[resamp])
result=logit_model.fit(method='lbfgs',maxiter=750)
print(result.summary())
```

executed in 4m 52s, finished 04:52:33 2021-04-22

```
                           Logit Regression Results
==============================================================================
Dep. Variable:            loan_status   No. Observations:              1138127
Model:                          Logit   Df Residuals:                  1138044
Method:                           MLE   Df Model:                           82
Date:                Thu, 22 Apr 2021   Pseudo R-squ.:                  0.4873
Time:                        04:52:33   Log-Likelihood:            -4.0445e+05
converged:                       True   LL-Null:                   -7.8889e+05
Covariance Type:            nonrobust   LLR p-value:                     0.000
================================================================================
========================
                                          coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
------------------------
loan_amnt                               2.1560      0.098     21.990      0.000
1.964       2.348
term                                    0.5027      0.016     30.586      0.000
0.470       0.535
int_rate                                7.2072      0.040    179.262      0.000
7.128       7.286
installment                            -1.5100      0.118    -12.783      0.000
-1.742      -1.279
emp_length                             -0.1368      0.007    -18.259      0.000
-0.151      -0.122
annual_inc                              0.3920      0.062      6.343      0.000
0.271       0.513
dti                                     2.6434      0.034     78.646      0.000
2.578       2.709
delinq_2yrs                             0.6637      0.019     34.573      0.000
0.626       0.701
inq_last_6mths                          0.3508      0.014     24.671      0.000
0.323       0.379
pub_rec                                 0.1738      0.015     11.632      0.000
0.144       0.203
revol_bal                              -1.6832      0.053    -31.759      0.000
-1.787      -1.579
total_acc                              -1.2758      0.034    -37.459      0.000
-1.343      -1.209
total_rec_late_fee                      4.4271      0.031    142.461      0.000
4.366       4.488
tot_cur_bal                            -0.2271      0.028     -8.237      0.000
-0.281      -0.173
bc_util                                 0.6411      0.025     25.833      0.000
0.592       0.690
mo_sin_rcnt_tl                          0.2385      0.024      9.827      0.000
0.191       0.286
mths_since_recent_bc                   -0.3742      0.022    -16.997      0.000
-0.417      -0.331
num_accts_ever_120_pd                   0.8164      0.020     40.442      0.000
0.777       0.856
num_bc_sats                             0.8718      0.035     24.604      0.000
```

```
0.802       0.941
num_bc_tl                                0.1275      0.032      3.952      0.000
0.064       0.191
num_op_rev_tl                            0.5645      0.032     17.836      0.000
0.502       0.627
num_tl_90g_dpd_24m                      -0.1595      0.030     -5.352      0.000
-0.218      -0.101
num_tl_op_past_12m                       1.0435      0.020     51.331      0.000
1.004       1.083
pct_tl_nvr_dlq                           1.3804      0.020     69.363      0.000
1.341       1.419
total_bal_ex_mort                        1.9785      0.077     25.700      0.000
1.828       2.129
total_bc_limit                          -1.3348      0.034    -38.920      0.000
-1.402      -1.268
total_il_high_credit_limit              -2.3481      0.069    -33.868      0.000
-2.484      -2.212
average_fico                            -0.8361      0.024    -35.528      0.000
-0.882      -0.790
sub_grade_A2                            -3.1012      0.032    -97.733      0.000
-3.163      -3.039
sub_grade_A3                            -3.2481      0.029   -110.253      0.000
-3.306      -3.190
sub_grade_A4                            -3.2478      0.023   -138.280      0.000
-3.294      -3.202
sub_grade_A5                            -3.2969      0.020   -165.388      0.000
-3.336      -3.258
sub_grade_B1                            -3.4009      0.018   -194.203      0.000
-3.435      -3.367
sub_grade_B2                            -3.6234      0.017   -214.474      0.000
-3.657      -3.590
sub_grade_B3                            -3.7443      0.016   -235.099      0.000
-3.776      -3.713
sub_grade_B4                            -3.8306      0.015   -253.739      0.000
-3.860      -3.801
sub_grade_B5                            -3.8679      0.015   -260.100      0.000
-3.897      -3.839
sub_grade_C1                            -3.9714      0.015   -271.353      0.000
-4.000      -3.943
sub_grade_C2                            -4.1335      0.015   -271.097      0.000
-4.163      -4.104
sub_grade_C3                            -4.1899      0.015   -271.874      0.000
-4.220      -4.160
sub_grade_C4                            -4.3151      0.016   -274.140      0.000
-4.346      -4.284
sub_grade_C5                            -4.5168      0.017   -271.297      0.000
-4.549      -4.484
sub_grade_D1                            -4.8100      0.019   -256.753      0.000
-4.847      -4.773
sub_grade_D2                            -5.0447      0.020   -252.285      0.000
-5.084      -5.006
sub_grade_D3                            -5.2398      0.021   -245.163      0.000
-5.282      -5.198
sub_grade_D4                            -5.4344      0.023   -238.138      0.000
-5.479      -5.390
sub_grade_D5                            -5.6902      0.025   -229.391      0.000
-5.739      -5.642
```

sub_grade_E1	-5.8298	0.028	-209.966	0.000	-5.884	-5.775
sub_grade_E2	-5.9535	0.029	-205.485	0.000	-6.010	-5.897
sub_grade_E3	-6.1471	0.031	-197.510	0.000	-6.208	-6.086
sub_grade_E4	-6.4066	0.034	-190.412	0.000	-6.473	-6.341
sub_grade_E5	-6.6574	0.035	-189.973	0.000	-6.726	-6.589
sub_grade_F1	-6.9581	0.042	-164.210	0.000	-7.041	-6.875
sub_grade_F2	-7.0097	0.049	-142.765	0.000	-7.106	-6.913
sub_grade_F3	-7.3057	0.054	-136.395	0.000	-7.411	-7.201
sub_grade_F4	-7.4520	0.060	-123.521	0.000	-7.570	-7.334
sub_grade_F5	-7.5491	0.065	-115.945	0.000	-7.677	-7.422
sub_grade_G1	-7.9231	0.074	-106.914	0.000	-8.068	-7.778
sub_grade_G2	-7.8289	0.090	-87.322	0.000	-8.005	-7.653
sub_grade_G3	-8.0357	0.099	-80.933	0.000	-8.230	-7.841
sub_grade_G4	-8.2587	0.110	-74.748	0.000	-8.475	-8.042
sub_grade_G5	-8.0126	0.119	-67.185	0.000	-8.246	-7.779
home_ownership_MORTGAGE	-0.2456	0.018	-13.527	0.000	-0.281	-0.210
home_ownership_OWN	-0.2941	0.019	-15.174	0.000	-0.332	-0.256
home_ownership_RENT	0.2025	0.018	11.450	0.000	0.168	0.237
verification_status_Source Verified	0.0042	0.007	0.631	0.528	-0.009	0.017
verification_status_Verified	-0.0982	0.008	-13.036	0.000	-0.113	-0.083
purpose_credit_card	-1.2692	0.013	-95.255	0.000	-1.295	-1.243
purpose_debt_consolidation	-0.9722	0.013	-77.556	0.000	-0.997	-0.948
purpose_home_improvement	-1.2495	0.018	-68.874	0.000	-1.285	-1.214
purpose_house	-1.3176	0.046	-28.556	0.000	-1.408	-1.227
purpose_major_purchase	-1.3732	0.026	-52.343	0.000	-1.425	-1.322
purpose_other	-1.2017	0.018	-68.096	0.000	-1.236	-1.167
purpose_renewable_energy	-1.3684	0.139	-9.843	0.000	-1.641	-1.096
purpose_small_business	-0.8205	0.033	-24.652	0.000	-0.886	-0.755
purpose_vacation	-1.3081	0.042	-31.443	0.000		

```
-1.390      -1.227
purpose_wedding                          -2.1062      0.169    -12.492      0.000
-2.437      -1.776
purpose_medical                          -1.1888      0.032    -36.988      0.000
-1.252      -1.126
purpose_moving                           -1.2438      0.040    -30.796      0.000
-1.323      -1.165
application_type_Joint App               -0.5429      0.018    -30.922      0.000
-0.577      -0.509
region_South                             -0.7420      0.007   -101.132      0.000
-0.756      -0.728
region_West                              -0.7465      0.008    -94.987      0.000
-0.762      -0.731
region_Northeast                         -0.6726      0.008    -79.360      0.000
-0.689      -0.656
==============================================================================
========================
```

In [73]:

```python
plt.rc('figure', figsize=(12, 25))
#plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 12}) old approach
plt.text(0.01, 0.05, str(result.summary()), {'fontsize': 12}, fontproperties = '
plt.axis('off')
plt.savefig('log_test.png')
```

executed in 2.24s, finished 06:44:14 2021-04-22

```
                           Logit Regression Results
==============================================================================
Dep. Variable:            loan_status   No. Observations:              1138127
Model:                          Logit   Df Residuals:                  1138044
Method:                           MLE   Df Model:                           82
Date:                Thu, 22 Apr 2021   Pseudo R-squ.:                  0.4873
Time:                        06:44:12   Log-Likelihood:            -4.0445e+05
converged:                       True   LL-Null:                   -7.8889e+05
Covariance Type:            nonrobust   LLR p-value:                     0.000
==================================================================================================
                                     coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
loan_amnt                          2.1560      0.098     21.990      0.000       1.964       2.348
term                               0.5027      0.016     30.586      0.000       0.470       0.535
int_rate                           7.2072      0.040    179.262      0.000       7.128       7.286
installment                       -1.5100      0.118    -12.783      0.000      -1.742      -1.279
emp_length                        -0.1368      0.007    -18.259      0.000      -0.151      -0.122
annual_inc                         0.3920      0.062      6.343      0.000       0.271       0.513
dti                                2.6434      0.034     78.646      0.000       2.578       2.709
delinq_2yrs                        0.6637      0.019     34.573      0.000       0.626       0.701
inq_last_6mths                     0.3508      0.014     24.671      0.000       0.323       0.379
pub_rec                            0.1738      0.015     11.632      0.000       0.144       0.203
revol_bal                         -1.6832      0.053    -31.759      0.000      -1.787      -1.579
total_acc                         -1.2758      0.034    -37.459      0.000      -1.343      -1.209
total_rec_late_fee                 4.4271      0.031    142.461      0.000       4.366       4.488
tot_cur_bal                       -0.2271      0.028     -8.237      0.000      -0.281      -0.173
bc_util                            0.6411      0.025     25.833      0.000       0.592       0.690
mo_sin_rcnt_tl                     0.2385      0.024      9.827      0.000       0.191       0.286
mths_since_recent_bc              -0.3742      0.022    -16.997      0.000      -0.417      -0.331
num_accts_ever_120_pd              0.8164      0.020     40.442      0.000       0.777       0.856
num_bc_sats                        0.8718      0.035     24.604      0.000       0.802       0.941
num_bc_tl                          0.1275      0.032      3.952      0.000       0.064       0.191
num_op_rev_tl                      0.5645      0.032     17.836      0.000       0.502       0.627
num_tl_90g_dpd_24m                -0.1595      0.030     -5.352      0.000      -0.218      -0.101
num_tl_op_past_12m                 1.0435      0.020     51.331      0.000       1.004       1.083
pct_tl_nvr_dlq                     1.3804      0.020     69.363      0.000       1.341       1.419
total_bal_ex_mort                  1.9785      0.077     25.700      0.000       1.828       2.129
total_bc_limit                    -1.3348      0.034    -38.920      0.000      -1.402      -1.268
total_il_high_credit_limit        -2.3481      0.069    -33.868      0.000      -2.484      -2.212
average_fico                      -0.8361      0.024    -35.528      0.000      -0.882      -0.790
sub_grade_A2                      -3.1012      0.032    -97.733      0.000      -3.163      -3.039
sub_grade_A3                      -3.2481      0.029   -110.253      0.000      -3.306      -3.190
sub_grade_A4                      -3.2478      0.023   -138.280      0.000      -3.294      -3.202
sub_grade_A5                      -3.2969      0.020   -165.388      0.000      -3.336      -3.258
sub_grade_B1                      -3.4009      0.018   -194.203      0.000      -3.435      -3.367
sub_grade_B2                      -3.6234      0.017   -214.474      0.000      -3.657      -3.590
sub_grade_B3                      -3.7443      0.016   -235.099      0.000      -3.776      -3.713
sub_grade_B4                      -3.8306      0.015   -253.739      0.000      -3.860      -3.801
sub_grade_B5                      -3.8679      0.015   -260.100      0.000      -3.897      -3.839
sub_grade_C1                      -3.9714      0.015   -271.353      0.000      -4.000      -3.943
sub_grade_C2                      -4.1335      0.015   -271.097      0.000      -4.163      -4.104
sub_grade_C3                      -4.1899      0.015   -271.874      0.000      -4.220      -4.160
sub_grade_C4                      -4.3151      0.016   -274.140      0.000      -4.346      -4.284
sub_grade_C5                      -4.5168      0.017   -271.297      0.000      -4.549      -4.484
sub_grade_D1                      -4.8100      0.019   -256.753      0.000      -4.847      -4.773
sub_grade_D2                      -5.0447      0.020   -252.285      0.000      -5.084      -5.006
sub_grade_D3                      -5.2398      0.021   -245.163      0.000      -5.282      -5.198
sub_grade_D4                      -5.4344      0.023   -238.138      0.000      -5.479      -5.390
sub_grade_D5                      -5.6902      0.025   -229.391      0.000      -5.739      -5.642
sub_grade_E1                      -5.8298      0.028   -209.966      0.000      -5.884      -5.775
sub_grade_E2                      -5.9535      0.029   -205.485      0.000      -6.010      -5.897
sub_grade_E3                      -6.1471      0.031   -197.510      0.000      -6.208      -6.086
sub_grade_E4                      -6.4066      0.034   -190.412      0.000      -6.473      -6.341
sub_grade_E5                      -6.6574      0.035   -189.973      0.000      -6.726      -6.589
sub_grade_F1                      -6.9581      0.042   -164.210      0.000      -7.041      -6.875
sub_grade_F2                      -7.0097      0.049   -142.765      0.000      -7.106      -6.913
sub_grade_F3                      -7.3057      0.054   -136.395      0.000      -7.411      -7.201
sub_grade_F4                      -7.4520      0.060   -123.521      0.000      -7.570      -7.334
```

```
sub_grade_F5                          -7.5491      0.065   -115.945      0.000      -7.677      -7.422
sub_grade_G1                          -7.9231      0.074   -106.914      0.000      -8.068      -7.778
sub_grade_G2                          -7.8289      0.090    -87.322      0.000      -8.005      -7.653
sub_grade_G3                          -8.0357      0.099    -80.933      0.000      -8.230      -7.841
sub_grade_G4                          -8.2587      0.110    -74.748      0.000      -8.475      -8.042
sub_grade_G5                          -8.0126      0.119    -67.185      0.000      -8.246      -7.779
home_ownership_MORTGAGE               -0.2456      0.018    -13.527      0.000      -0.281      -0.210
home_ownership_OWN                    -0.2941      0.019    -15.174      0.000      -0.332      -0.256
home_ownership_RENT                    0.2025      0.018     11.450      0.000       0.168       0.237
verification_status_Source Verified    0.0042      0.007      0.631      0.528      -0.009       0.017
verification_status_Verified          -0.0982      0.008    -13.036      0.000      -0.113      -0.083
purpose_credit_card                   -1.2692      0.013    -95.255      0.000      -1.295      -1.243
purpose_debt_consolidation            -0.9722      0.013    -77.556      0.000      -0.997      -0.948
purpose_home_improvement              -1.2495      0.018    -68.874      0.000      -1.285      -1.214
purpose_house                         -1.3176      0.046    -28.556      0.000      -1.408      -1.227
purpose_major_purchase                -1.3732      0.026    -52.343      0.000      -1.425      -1.322
purpose_other                         -1.2017      0.018    -68.096      0.000      -1.236      -1.167
purpose_renewable_energy              -1.3684      0.139     -9.843      0.000      -1.641      -1.096
purpose_small_business                -0.8205      0.033    -24.652      0.000      -0.886      -0.755
purpose_vacation                      -1.3081      0.042    -31.443      0.000      -1.390      -1.227
purpose_wedding                       -2.1062      0.169    -12.492      0.000      -2.437      -1.776
purpose_medical                       -1.1888      0.032    -36.988      0.000      -1.252      -1.126
purpose_moving                        -1.2438      0.040    -30.796      0.000      -1.323      -1.165
application_type_Joint App            -0.5429      0.018    -30.922      0.000      -0.577      -0.509
region_South                          -0.7420      0.007   -101.132      0.000      -0.756      -0.728
region_West                           -0.7465      0.008    -94.987      0.000      -0.762      -0.731
region_Northeast                      -0.6726      0.008    -79.360      0.000      -0.689      -0.656
=========================================================================================================
```

In [ ]:

```python
#smote log reg without recoveries
logreg = LogisticRegression(C=.75,fit_intercept=False,solver='saga',penalty='l1',
log_model = logreg.fit(x_train_final_z[resamp], y_train_final_z.loan_status)
y_hat_log = logreg.predict(x_test_scaled_z[resamp])
y_score_log = log_model.decision_function(x_test_scaled_z[resamp])
fpr,tpr,thresholds = roc_curve(y_test_z,y_score_log)

print('AUC: {}'.format(auc(fpr, tpr)))
cf = confusion_matrix(y_test_z,y_hat_log)

plot_confusion_matrix(log_model,x_test_scaled_z[resamp],y_test_z,cmap=plt.cm.Blue
                      display_labels=["Fully Paid", "Charged Off"],
                       values_format=".5g")
plt.title("Logistic Regression Confusion Matrix")

print(confusion_matrix(y_test_z, y_hat_log))
print(classification_report(y_test_z, y_hat_log))
```

In [67]:

```python
log_summary = model_scores(y_test_z,y_hat_log,"Logistic Regression")
log_summary
```

executed in 403ms, finished 05:11:37 2021-04-22

Out[67]:

	Model	precision_score	recall_score	accuracy_score	f1_score
0	Logistic Regression	0.408908	0.236125	0.797429	0.299375

In [68]:

```python
log_summary.to_csv('data/log_scores')
```

executed in 31ms, finished 05:12:12 2021-04-22

In [65]:

```
def model_scores(y_true,y_pred,model_name):
    results = ({'Model':model_name,
                'precision_score': precision_score(y_true,y_pred),
                'recall_score': recall_score(y_true,y_pred),
                'accuracy_score': accuracy_score(y_true,y_pred),
                'f1_score': f1_score(y_true,y_pred)})
    model_results = pd.DataFrame(data=results,index=[0])
    return model_results
```

executed in 6ms, finished 05:11:30 2021-04-22