INTELLIGENT INVESTMENT CONSULTANT FOR LENDING CLUB</p>

Author: Kun Jiang

Email: kunjiang0112@gmail.com

TABLE OF CONTENTS

INTRODUCTION

[go back to the top]

Lending Club (https://www.lendingclub.com/) is a US peer-to-peer lending company that provides a largest peer-to-peer online lending platform. The Lending Club's platform enables borrowers to obtain a loan funded by investors. Investors profit from interest payments on loans. Based on the levels of loans' default rate, Lending categorizes the quality of the loans into seven grades, each of which assigned with different interest rates and risks. From the investors' point of view, they want to find out the potential default rate of each specific loans, in order to avoid investing on those loans with high risk of default. **The goal of this project is to build a predictive model to predict the default rate of the newly issued loans, based on information from historic loans.**

The centerpiece of this model's operation is to use algorithm to choose which notes to invest in. Often investors have to choose between hundreds or thousands of available loans at Lending Club. This model makes the process easier by using machine-learning to calculate which notes are more likely to perform better than others. The moment new loans are added to the platforms, the algorithm analyzes the variables of these loans and only invests in the best ones.

PROJECT SUMMARY

[go back to the top]

This project is implemented using Python on the Jupyter Notebook platform. Data processing and machine learning packages such as Pandas and Sci-kit Learn are utilized. The modeling is trained based on the Lending Club's historic loan data from year 2014.

The target is to predict the default rate of the new loans listed on Lending Club. The workflow of the project consists of 6 steps:

Data cleaning are performed to uniform the features used in the training set and the target set. Categorical variables are extracted into meaningful features for model input via feature engineering. XGBoost (Extreme Gradient Boost) is used as the modeling algorithm. As an advanced Gradient Boosted Regression Trees (GBRT) algorithm, XGBoost adopted several advantages of GBRT such as: natural handling of data of mixed type , predictive power, robustness to outliers in output space (via robust loss functions). Combining both Random Forest and Adaptive Gradient Boosting algorithms, XGBoost exhibited high efficiency and flexibility in modeling large datasets with ease of implementation.

The model is then autotuned to optimize its performance using Bayesian Optimization and Cross Validation. The tuned model is tested using the Area Under Curve (AUC) of the Receiver Operating Characteristics (ROC) as the performance metrics. The overall AUC score reaches above 73% on the testing set. Finally, the tuned model is saved using Pickle and integrated into an interactive web-application built with Flask Framework.

The total runtime for model building and model tuning may take 4-5 hours, depending on computer hardware.

Required Libraries

This notebook uses several standard Python packages and special packages for model building and model tuning. The primary libraries that we'll be using are:

- **requests**: request data from API servers.
- **pandas**: and **numpy**: numeric paneled data operation.
- **matplotlib** and **seaborn**: data visualization.
- **bayes_opt**: Bayesian Optimization.
- **sklearn**: Sci-kit Learn for model buidling and evaluation.
- **xgboost**: XGBoost algorithm.

In [1]:
```
import requests
import json
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import datetime as dt
import seaborn as sns
import xgboost as xgb
%matplotlib inline
```

Data Acquisition

[go back to the top]

Load Current Loan Data

Lending Club constantly updates its newly issued loans, which are accessed via Lending Club's API services. The data are loaded into Pandas DataFrame using Python's "requests" function. The current loan data are then saved into json format and read into Pandas DataFrame.

In [2]:
```
api_key = open('apikeys.txt', 'r').read()
headers = {"Authorization": api_key}
url = 'https://api.lendingclub.com/api/investor/v1/loans/listing'
```

In [3]:
```
r = requests.get(url, headers = headers)
```

In [4]:
```
data = r.json()
```

In [5]:
```
myData = data['loans']
```

In [6]:
```
with open('current_list.txt', 'w') as outfile:
    json.dump(myData, outfile)
```

In [7]:
```
df_current = pd.read_json('current_list.txt')
```

In [8]:
```
df_current.head()
```

Out[8]:

	accNowDelinq	accOpenPast24Mths	acceptD	addrState	addrZip	allUtil	annualInc	ar
0	0	2	2018-06-16T13:52:44.000-07:00	NY	145xx	50.5	74000	N
1	0	8	2018-06-16T16:51:35.000-07:00	IL	612xx	1.0	90000	N

2 rows × 119 columns

In [9]:
```
df_current.shape
```

Out[9]:
```
(2, 119)
```

In [10]:
```
print(df_current.columns.values)
```

```
['accNowDelinq' 'accOpenPast24Mths' 'acceptD' 'addrState' 'addrZip'
 'allUtil' 'annualInc' 'annualIncJoint' 'applicationType' 'avgCurBal'
 'bcOpenToBuy' 'bcUtil' 'chargeoffWithin12Mths' 'collections12MthsExMed'
 'creditPullD' 'delinq2Yrs' 'delinqAmnt' 'desc' 'disbursementMethod' 'dti'
 'dtiJoint' 'earliestCrLine' 'empLength' 'empTitle' 'expD' 'expDefaultRate'
 'ficoRangeHigh' 'ficoRangeLow' 'fundedAmount' 'grade' 'homeOwnership'
 'housingPayment' 'iLUtil' 'id' 'ilsExpD' 'initialListStatus' 'inqFi'
 'inqLast12m' 'inqLast6Mths' 'installment' 'intRate' 'investorCount'
 'isIncV' 'isIncVJoint' 'listD' 'loanAmount' 'maxBalBc' 'memberId'
 'moSinOldIlAcct' 'moSinOldRevTlOp' 'moSinRcntRevTlOp' 'moSinRcntTl'
 'mortAcc' 'mtgPayment' 'mthsSinceLastDelinq' 'mthsSinceLastMajorDerog'
 'mthsSinceLastRecord' 'mthsSinceRcntIl' 'mthsSinceRecentBc'
 'mthsSinceRecentBcDlq' 'mthsSinceRecentInq' 'mthsSinceRecentRevolDelinq'
 'numAcctsEver120Ppd' 'numActvBcTl' 'numActvRevTl' 'numBcSats' 'numBcTl'
 'numIlTl' 'numOpRevTl' 'numRevAccts' 'numRevTlBalGt0' 'numSats'
 'numTl120dpd2m' 'numTl30dpd' 'numTl90gDpd24m' 'numTlOpPast12m' 'openAcc'
 'openAcc6m' 'openActIl' 'openIl12m' 'openIl24m' 'openRv12m' 'openRv24m'
 'pctTlNvrDlq' 'percentBcGt75' 'pubRec' 'pubRecBankruptcies' 'purpose'
 'reviewStatus' 'reviewStatusD' 'revolBal' 'revolBalJoint' 'revolUtil'
 'secAppChargeoffWithin12Mths' 'secAppCollections12MthsExMed'
 'secAppEarliestCrLine' 'secAppFicoRangeHigh' 'secAppFicoRangeLow'
 'secAppInqLast6Mths' 'secAppMortAcc' 'secAppMthsSinceLastMajorDerog'
 'secAppNumRevAccts' 'secAppOpenAcc' 'secAppOpenActIl' 'secAppRevolUtil'
 'serviceFeeRate' 'subGrade' 'taxLiens' 'term' 'totCollAmt' 'totCurBal'
 'totHiCredLim' 'totalAcc' 'totalBalExMort' 'totalBalIl' 'totalBcLimit'
 'totalCuTl' 'totalIlHighCreditLimit' 'totalRevHiLim']
```

Load 2014 Loan Data

Lending Club offers open access to its historical loan data from 2007. In this project the loan data from year 2014 were loaded into Pandas DataFrame to be used as the training set.

In [11]:
```
df_2014 = pd.read_csv('data/LoanStats2014.csv', skiprows = 1, skipfooter=2)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: ParserWarning: Fallin
g back to the 'python' engine because the 'c' engine does not support skipfooter; you
can avoid this warning by specifying engine='python'.
  """Entry point for launching an IPython kernel.
```

In [12]:
```
df_2014.head()
```

Out[12]:

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installmen
0	36805548	NaN	10400	10400	10400	36 months	6.99%	321.08
1	38098114	NaN	15000	15000	15000	60 months	12.39%	336.64
2	37822187	NaN	9600	9600	9600	36 months	13.66%	326.53
3	37662224	NaN	7650	7650	7650	36 months	13.66%	260.20
4	37612354	NaN	12800	12800	12800	60 months	17.14%	319.08

5 rows × 128 columns

In [13]:
```
df_2014.shape
```

Out[13]: (235629, 128)

In [14]:
```
print(df_2014.columns.values)
```

```
['id' 'member_id' 'loan_amnt' 'funded_amnt' 'funded_amnt_inv' 'term'
 'int_rate' 'installment' 'grade' 'sub_grade' 'emp_title' 'emp_length'
 'home_ownership' 'annual_inc' 'verification_status' 'issue_d'
 'loan_status' 'pymnt_plan' 'url' 'desc' 'purpose' 'title' 'zip_code'
 'addr_state' 'dti' 'delinq_2yrs' 'earliest_cr_line' 'fico_range_low'
 'fico_range_high' 'inq_last_6mths' 'mths_since_last_delinq'
 'mths_since_last_record' 'open_acc' 'pub_rec' 'revol_bal' 'revol_util'
 'total_acc' 'initial_list_status' 'out_prncp' 'out_prncp_inv'
 'total_pymnt' 'total_pymnt_inv' 'total_rec_prncp' 'total_rec_int'
 'total_rec_late_fee' 'recoveries' 'collection_recovery_fee' 'last_pymnt_d'
 'last_pymnt_amnt' 'next_pymnt_d' 'last_credit_pull_d'
 'last_fico_range_high' 'last_fico_range_low' 'collections_12_mths_ex_med'
 'mths_since_last_major_derog' 'policy_code' 'application_type'
 'annual_inc_joint' 'dti_joint' 'verification_status_joint'
 'acc_now_delinq' 'tot_coll_amt' 'tot_cur_bal' 'open_acc_6m' 'open_il_6m'
 'open_il_12m' 'open_il_24m' 'mths_since_rcnt_il' 'total_bal_il' 'il_util'
 'open_rv_12m' 'open_rv_24m' 'max_bal_bc' 'all_util' 'total_rev_hi_lim'
 'inq_fi' 'total_cu_tl' 'inq_last_12m' 'acc_open_past_24mths' 'avg_cur_bal'
 'bc_open_to_buy' 'bc_util' 'chargeoff_within_12_mths' 'delinq_amnt'
 'mo_sin_old_il_acct' 'mo_sin_old_rev_tl_op' 'mo_sin_rcnt_rev_tl_op'
 'mo_sin_rcnt_tl' 'mort_acc' 'mths_since_recent_bc'
 'mths_since_recent_bc_dlq' 'mths_since_recent_inq'
 'mths_since_recent_revol_delinq' 'num_accts_ever_120_pd' 'num_actv_bc_tl'
 'num_actv_rev_tl' 'num_bc_sats' 'num_bc_tl' 'num_il_tl' 'num_op_rev_tl'
 'num_rev_accts' 'num_rev_tl_bal_gt_0' 'num_sats' 'num_tl_120dpd_2m'
 'num_tl_30dpd' 'num_tl_90g_dpd_24m' 'num_tl_op_past_12m' 'pct_tl_nvr_dlq'
 'percent_bc_gt_75' 'pub_rec_bankruptcies' 'tax_liens' 'tot_hi_cred_lim'
 'total_bal_ex_mort' 'total_bc_limit' 'total_il_high_credit_limit'
 'revol_bal_joint' 'sec_app_fico_range_low' 'sec_app_fico_range_high'
 'sec_app_earliest_cr_line' 'sec_app_inq_last_6mths' 'sec_app_mort_acc'
 'sec_app_open_acc' 'sec_app_revol_util' 'sec_app_open_il_6m'
 'sec_app_num_rev_accts' 'sec_app_chargeoff_within_12_mths'
 'sec_app_collections_12_mths_ex_med' 'sec_app_mths_since_last_major_derog']
```

Exploratory Analysis

[go back to the top]

Exploratory Analysis consist of parts:

In [15]:
```
df_current.isnull().sum()
```

Out[15]:
```
accNowDelinq                  0
accOpenPast24Mths             0
acceptD                       0
addrState                     0
addrZip                       0
allUtil                       0
annualInc                     0
annualIncJoint                2
applicationType               0
avgCurBal                     0
bcOpenToBuy                   0
bcUtil                        0
chargeoffWithin12Mths         0
collections12MthsExMed        0
creditPullD                   0
delinq2Yrs                    0
delinqAmnt                    0
desc                          2
disbursementMethod            0
dti                           0
dtiJoint                      2
earliestCrLine                0
empLength                     0
empTitle                      0
expD                          0
expDefaultRate                0
ficoRangeHigh                 0
ficoRangeLow                  0
fundedAmount                  0
grade                         0
                            ..
reviewStatusD                 0
revolBal                      0
revolBalJoint                 2
revolUtil                     0
secAppChargeoffWithin12Mths   2
secAppCollections12MthsExMed  2
secAppEarliestCrLine          2
secAppFicoRangeHigh           2
secAppFicoRangeLow            2
secAppInqLast6Mths            2
secAppMortAcc                 2
secAppMthsSinceLastMajorDerog 2
secAppNumRevAccts             2
secAppOpenAcc                 2
secAppOpenActIl               2
secAppRevolUtil               2
serviceFeeRate                0
subGrade                      0
taxLiens                      0
term                          0
totCollAmt                    0
totCurBal                     0
totHiCredLim                  0
totalAcc                      0
totalBalExMort                0
totalBalIl                    0
totalBcLimit                  0
totalCuTl                     0
totalIlHighCreditLimit        0
totalRevHiLim                 0
Length: 119, dtype: int64
```

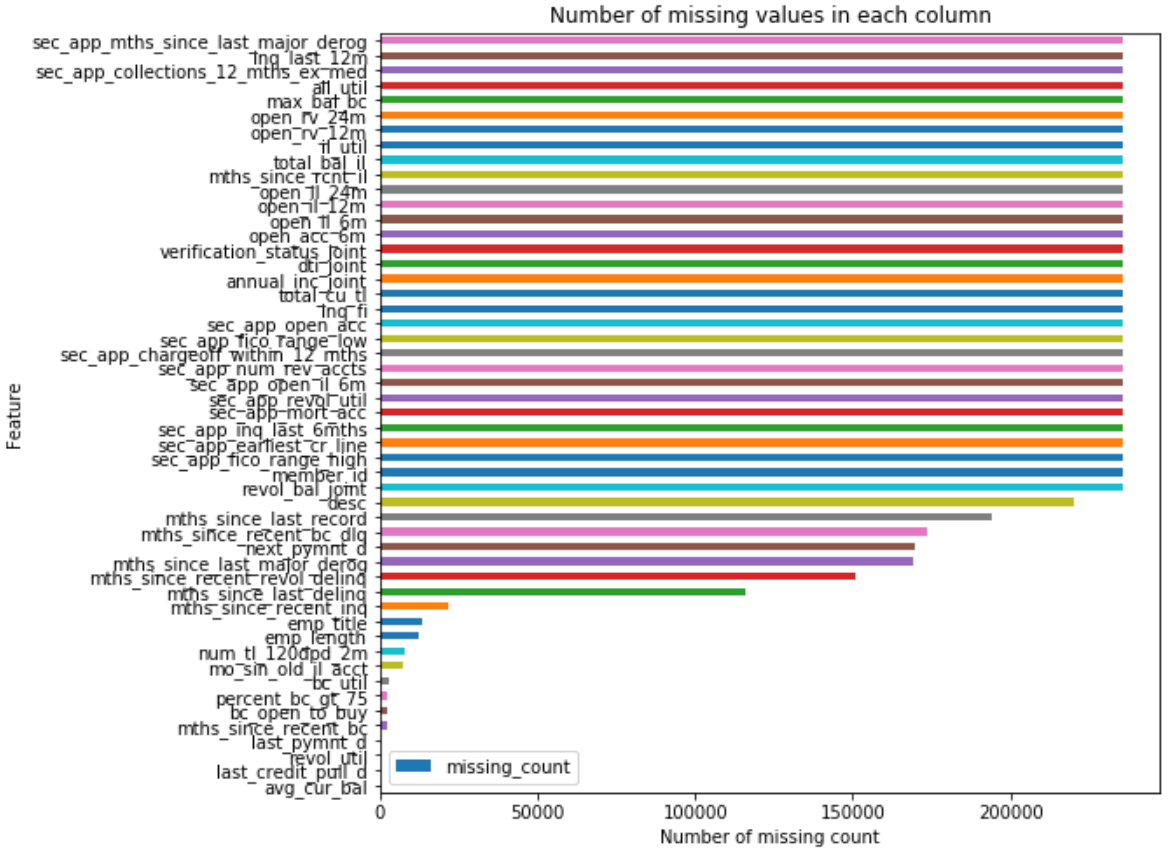
In [16]:
```
df_2014.isnull().sum()
```

Out[16]:
```
id                                      0
member_id                          235629
loan_amnt                               0
funded_amnt                             0
funded_amnt_inv                         0
term                                    0
int_rate                                0
installment                             0
grade                                   0
sub_grade                               0
emp_title                           13236
emp_length                          12019
home_ownership                          0
annual_inc                              0
verification_status                     0
issue_d                                 0
loan_status                             0
pymnt_plan                              0
url                                     0
desc                               220350
purpose                                 0
title                                   0
zip_code                                0
addr_state                              0
dti                                     0
delinq_2yrs                             0
earliest_cr_line                        0
fico_range_low                          0
fico_range_high                         0
inq_last_6mths                          0
                                      ...
num_il_tl                               0
num_op_rev_tl                           0
num_rev_accts                           0
num_rev_tl_bal_gt_0                     0
num_sats                                0
num_tl_120dpd_2m                     7860
num_tl_30dpd                            0
num_tl_90g_dpd_24m                      0
num_tl_op_past_12m                      0
pct_tl_nvr_dlq                          0
percent_bc_gt_75                     2557
pub_rec_bankruptcies                    0
tax_liens                               0
tot_hi_cred_lim                         0
total_bal_ex_mort                       0
total_bc_limit                          0
total_il_high_credit_limit              0
revol_bal_joint                    235629
sec_app_fico_range_low             235629
sec_app_fico_range_high            235629
sec_app_earliest_cr_line           235629
sec_app_inq_last_6mths             235629
sec_app_mort_acc                   235629
sec_app_open_acc                   235629
sec_app_revol_util                 235629
sec_app_open_il_6m                 235629
sec_app_num_rev_accts              235629
sec_app_chargeoff_within_12_mths   235629
sec_app_collections_12_mths_ex_med 235629
sec_app_mths_since_last_major_derog 235629
Length: 128, dtype: int64
```

Generate a dataframe called missing_df from df_train, in which there are two columns, one is the column names of our features, the other column is the missing_count (the number of missing values) of that feature. The table should be ordered by missing_count decendingly.

In [17]:
```
missing_df = df_2014.isnull().sum().reset_index()
missing_df.columns = ['feature_name', 'missing_count']
missing_df = missing_df.loc[missing_df['missing_count']>0]
missing_df = missing_df.sort_values(by='missing_count')
```

In [18]:
```
missing_df.plot(x='feature_name', y='missing_count', kind='barh', figsize=(8,8),fonts
ize = 10, logx=False )
plt.title("Number of missing values in each column", fontsize = 12)
plt.xlabel("Number of missing count",fontsize=10)
plt.ylabel("Feature",fontsize=10)
plt.show()
```



In [19]:
```
df_2014= df_2014.dropna(axis =1, how = 'all')
```

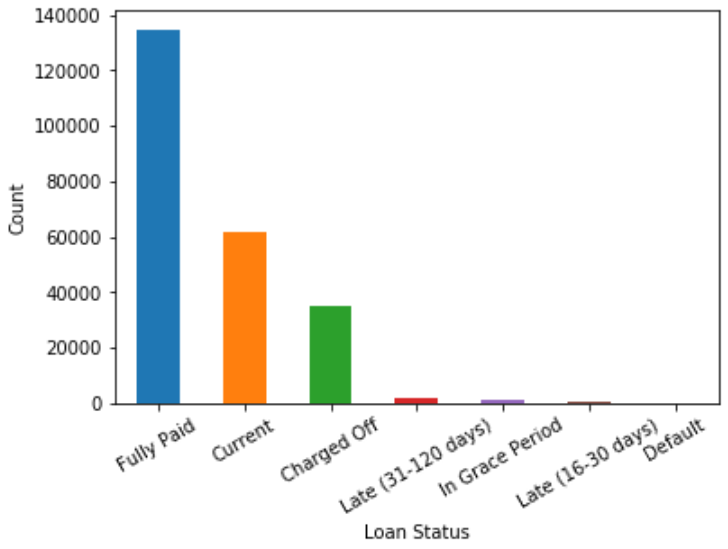
In [20]:
```
df_2014.shape, df_current.shape
```

Out[20]:
```
((235629, 97), (2, 119))
```

The target feature of the predictive model is labeled as "loan status", which refers to the stutus of loans as of today. A histogram is plotted to visualize the counts of different statuses. There are totally 7 different loan statuses, among which "Fully Paid" and "Charged Off" refer to those loans whose statuses are already determined, while all other statuses may still change over time. In this project only the loans with statuses of "Fully Paid" and "Charged Off" are considered. They are labeled as "0" (no default) and "1" (default), respectively.

In [21]:
```
fig = df_2014.loan_status.value_counts().plot('bar', rot = 90)
fig.set_xlabel('Loan Status')
fig.set_ylabel('Count')
plt.xticks(rotation = 30)
```

Out[21]: (array([0, 1, 2, 3, 4, 5, 6]), <a list of 7 Text xticklabel objects>)

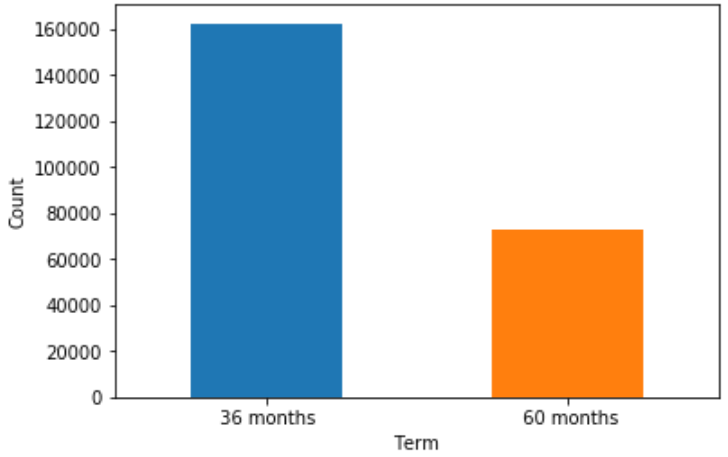


From the above histogram, it is seen that the ratio between "Fully Paid" and "Charged Off" loans are about 4:1. Discarding all other loan statuses, the overall default rate for the 2014 loans is about 0.2. Over time, this ratio may change slightly but this does not influence the training results much.

The term of the loans has only two values, 36 months or 60 months, in the ratio of about 2:1. As of the date of this project, most of the 36 months loans should be closed but most 60 months loans are still open. So this model will have a better prediction performance on the 36 months loans than the 60 months loans.

In [22]:
```
fig = df_2014.term.value_counts().plot('bar', rot = 0)
fig.set_xlabel('Term')
fig.set_ylabel('Count')
```

Out[22]: Text(0,0.5,'Count')



Feature Exploration

In [23]:
```
df_2014.shape, df_current.shape
```

Out[23]: ((235629, 97), (2, 119))

Comparing the features between the historic and the current data, there are several distinct differences: First, the number of features in the 2014 data is more than the current data. Second, the feature names are labeled differently: the historic feature names are separated by underscore "_", while the current feature names have upper case letters. To uniform the feature names, the underscores are deleted in the historic data, and the letters are all converted to lower cases in the current data. In some cases, different feature names may refer to the same variable, so those feature names are uniformed also.

Features names have been formatted using the same rule before usage

In [24]:
```
cols = []
for col in df_2014.columns:
    cols.append(col.replace("_", ""))
```

In [25]:
```
df_2014.columns = cols
```

In [26]:
```
cols = []
for col in df_current.columns:
    cols.append(col.lower())
```

In [27]:
```
df_current.columns = cols
```

In [28]:
```
# Change some column names in df_2014 to match df_current
# Fundedamnt to fundedamount. But this feature is not used since all loans in history
 data are fully funded.
df_2014.rename(columns={'zipcode': 'addrzip', 'verificationstatus': 'isincv', 'verifi
cationstatusjoint': 'isincvjoint',\
                        'loanamnt': 'loanamount', 'numacctsever120pd': 'numacctsever1
20ppd'}, inplace=True)
```

Having uniformed the feature names, the next task is to find the common features in both historic and current data sets. The common features are extracted using set operations. The historic data frame df_2014 is then reduced to only have the common features plus the loan status ("loanstatus") and issued date ("issued").

In [29]:
```
#Find common columns
com_cols = set(df_2014.columns) & set(df_current.columns)
hist_unique = set(df_2014.columns) - set (df_current.columns)
cur_unique = set(df_current.columns) - set (df_2014.columns)
```

In [30]:
```
print(len(com_cols))
print(com_cols)
```

```
74
{'applicationtype', 'numtloppast12m', 'mosinoldrevtlop', 'numiltl', 'bcopentobuy', 'd
esc', 'ficorangelow', 'avgcurbal', 'id', 'mosinrcntrevtlop', 'numtl90gdpd24m', 'loana
mount', 'pubrec', 'totcollamt', 'earliestcrline', 'numbctl', 'numrevaccts', 'totcurba
l', 'mthssincelastrecord', 'mthssincerecentbcdlq', 'totalrevhilim', 'revolutil', 'tot
alacc', 'initialliststatus', 'accnowdelinq', 'numbcsats', 'mthssincerecentinq', 'mths
sincelastdelinq', 'purpose', 'ficorangehigh', 'mthssincelastmajorderog', 'annualinc',
'mthssincerecentrevoldelinq', 'totalilhighcreditlimit', 'numoprevtl', 'numtl120dpd2
m', 'numacctsever120ppd', 'numactvrevtl', 'chargeoffwithin12mths', 'totalbclimit', 'n
umsats', 'isincv', 'mthssincerecentbc', 'mosinoldilacct', 'emptitle', 'mortacc', 'ins
tallment', 'numrevtlbalgt0', 'dti', 'numactvbctl', 'collections12mthsexmed', 'taxlien
s', 'bcutil', 'percentbcgt75', 'term', 'addrstate', 'mosinrcnttl', 'numtl30dpd', 'tot
hicredlim', 'homeownership', 'openacc', 'subgrade', 'pcttlnvrdlq', 'totalbalexmort',
'delinqamnt', 'revolbal', 'inqlast6mths', 'accopenpast24mths', 'pubrecbankruptcies',
'emplength', 'grade', 'intrate', 'delinq2yrs', 'addrzip'}
```

In [31]:
```
print(len(hist_unique))
print(hist_unique)
```

```
23
{'recoveries', 'totalpymntinv', 'pymntplan', 'lastficorangehigh', 'outprncp', 'loanst
atus', 'outprncpinv', 'lastpymntamnt', 'totalpymnt', 'fundedamntinv', 'lastcreditpull
d', 'title', 'lastpymntd', 'collectionrecoveryfee', 'url', 'nextpymntd', 'totalrecprn
cp', 'issued', 'totalrecint', 'fundedamnt', 'totalreclatefee', 'policycode', 'lastfic
orangelow'}
```

In [32]:
```
print(len(cur_unique))
print(cur_unique)
```

```
45
{'annualincjoint', 'investorcount', 'secappmthssincelastmajorderog', 'secappcollectio
ns12mthsexmed', 'dtijoint', 'housingpayment', 'secappficorangelow', 'openacc6m', 'mem
berid', 'expd', 'maxbalbc', 'totalcutl', 'secappopenactil', 'expdefaultrate', 'inqf
i', 'openrv12m', 'reviewstatus', 'secappnumrevaccts', 'mthssincercntil', 'secappearli
estcrline', 'allutil', 'secappmortacc', 'secappopenacc', 'secappficorangehigh', 'seca
ppchargeoffwithin12mths', 'isincvjoint', 'creditpulld', 'listd', 'reviewstatusd', 'ac
ceptd', 'servicefeerate', 'secapprevolutil', 'openil12m', 'revolbaljoint', 'openacti
l', 'openil24m', 'ilsexpd', 'inqlast12m', 'disbursementmethod', 'fundedamount', 'open
rv24m', 'secappinqlast6mths', 'mtgpayment', 'totalbalil', 'ilutil'}
```

In [33]:
```
df_2014 = df_2014[list(com_cols) + ["loanstatus", "issued"]]
```

Data Exploration

The next step is the preliminary data exploration. There are many all-null columns, which are dropped from the training set. The rows with loan statuses other than "Charged Off" or "Fully Paid" are also dropped from the training set. The resulted data frame has three different data types: "object", "int64", and "float64". All the "object" features are saved into a list of categorical variables "cat_vars', and all the numeric features are saved into "num_vars".

There are 17 resulted categorical features:

['intrate', 'addrzip', 'emplength', 'initialliststatus', 'emptitle', 'earliestcrline', 'grade', 'isincv', 'revolutil', 'homeownership', 'term', 'subgrade', 'purpose', 'addrstate', 'applicationtype', 'loanstatus', 'issued']

Each of them will be converted or transformed in the Feature Engineering step to be processed by the model algorithm.

In [34]:
```
df_2014 = df_2014[df_2014.loanstatus.isin(["Charged Off", "Fully Paid"])]
```

In [35]:
```
df_2014_raw = df_2014
```

In [36]:
```
df_2014.shape
```

Out[36]: (169475, 76)

In [37]:
```
data_types = df_2014.dtypes
cat_vars = list(data_types[data_types=='object'].index)
num_vars = list(data_types[data_types=='int64'].index) + list(data_types[data_types==
'float64'].index)
```

In [38]:
```
print ("There are %d categorical features. We will process them one by one." %len(cat
_vars))
```

```
There are 18 categorical features. We will process them one by one.
```

In [39]:
```
print(cat_vars)
```

```
['applicationtype', 'desc', 'earliestcrline', 'revolutil', 'initialliststatus', 'purp
ose', 'isincv', 'emptitle', 'term', 'addrstate', 'homeownership', 'subgrade', 'emplen
gth', 'grade', 'intrate', 'addrzip', 'loanstatus', 'issued']
```

In [40]:
```
print ("There are %d numerical features" %len(num_vars))
```

```
There are 58 numerical features
```

In [41]:
```
print(num_vars)
```

```
['numtloppast12m', 'mosinoldrevtlop', 'numiltl', 'ficorangelow', 'id', 'mosinrcntrevt
lop', 'numtl90gdpd24m', 'loanamount', 'pubrec', 'totcollamt', 'numbctl', 'numrevacct
s', 'totcurbal', 'totalrevhilim', 'totalacc', 'accnowdelinq', 'numbcsats', 'ficorange
high', 'totalilhighcreditlimit', 'numoprevtl', 'numacctsever120ppd', 'numactvrevtl',
'chargeoffwithin12mths', 'totalbclimit', 'numsats', 'mortacc', 'numrevtlbalgt0', 'num
actvbctl', 'collections12mthsexmed', 'taxliens', 'mosinrcnttl', 'numtl30dpd', 'tothic
redlim', 'openacc', 'totalbalexmort', 'delinqamnt', 'revolbal', 'inqlast6mths', 'acco
penpast24mths', 'pubrecbankruptcies', 'delinq2yrs', 'bcopentobuy', 'avgcurbal', 'mths
sincelastrecord', 'mthssincerecentbcdlq', 'mthssincerecentinq', 'mthssincelastdelin
q', 'mthssincelastmajorderog', 'annualinc', 'mthssincerecentrevoldelinq', 'numtl120dp
d2m', 'mthssincerecentbc', 'mosinoldilacct', 'installment', 'dti', 'bcutil', 'percent
bcgt75', 'pcttlnvrdlq']
```

Exploratory Visualization

Visualization packages "Matplotlib" and "Seaborn" were used to plot the distribution of the categorical features.

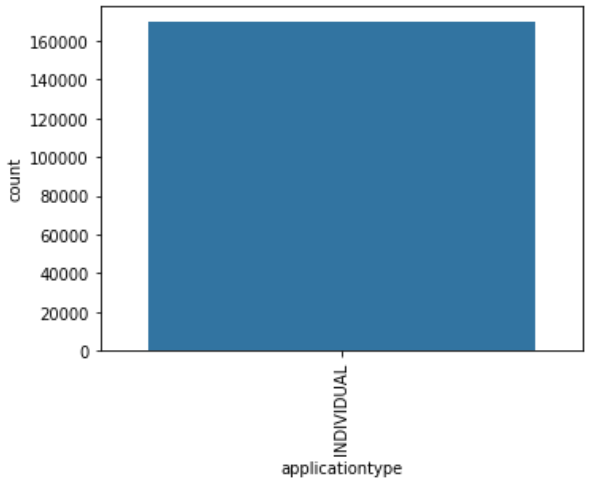
In [42]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

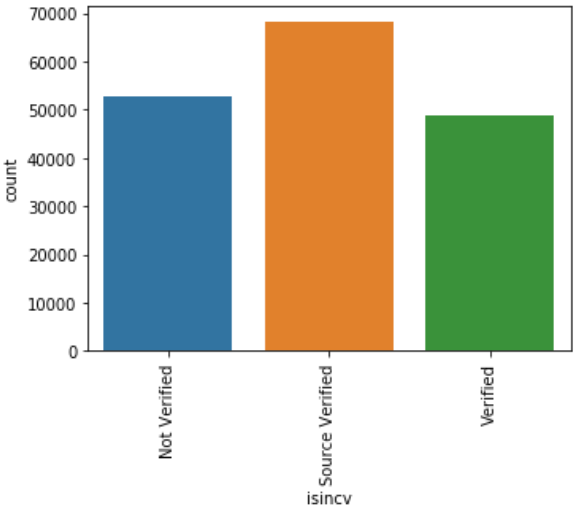
```
18 58 (169475, 76)
```

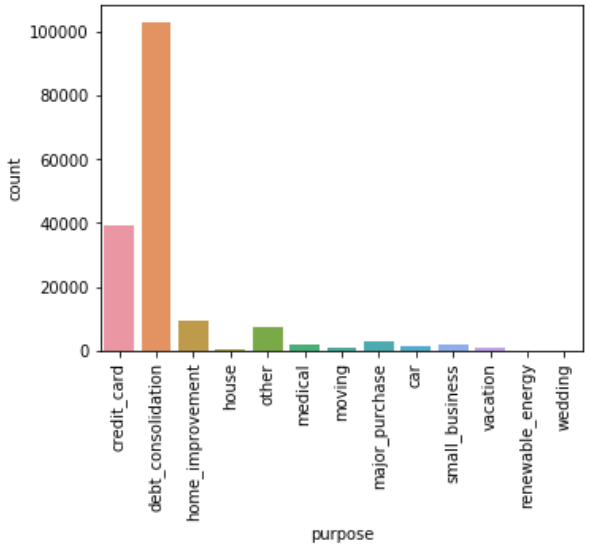
In [43]:
```
cat_vars_plot = ['applicationtype', 'isincv', 'purpose', 'subgrade', 'initialliststat
us', 'addrstate', 'grade', 'term',\
                 'homeownership', 'emplength']
```

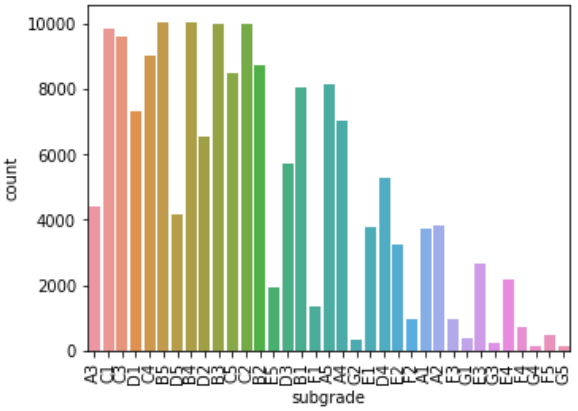
In [44]:
```
n_cols = 2
n_rows = 5
for i in range(n_rows):
    fg,ax = plt.subplots(nrows=1,ncols=n_cols,figsize=(12, 4))
    for j in range(n_cols):
        g = sns.countplot(x=cat_vars_plot[i*n_cols+j], data=df_2014, ax=ax[j])
        g.set_xticklabels(labels=g.get_xticklabels(), rotation = 90)
```

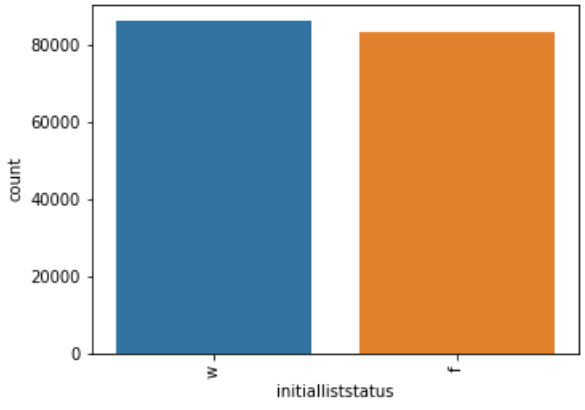
```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

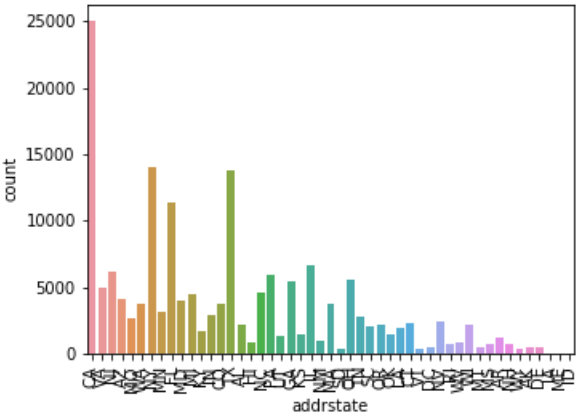


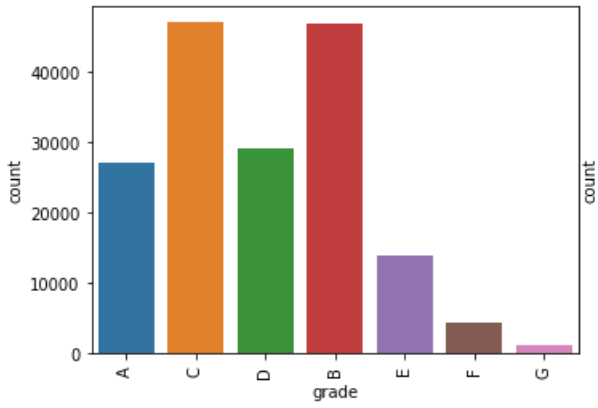


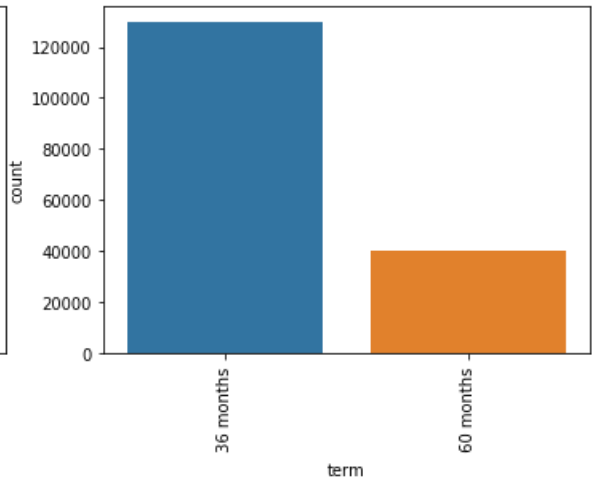


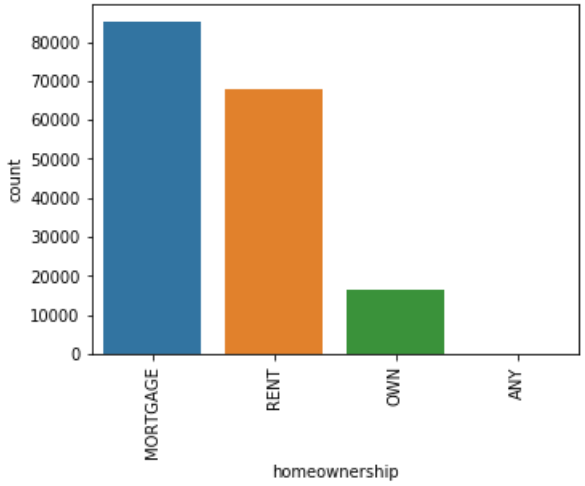


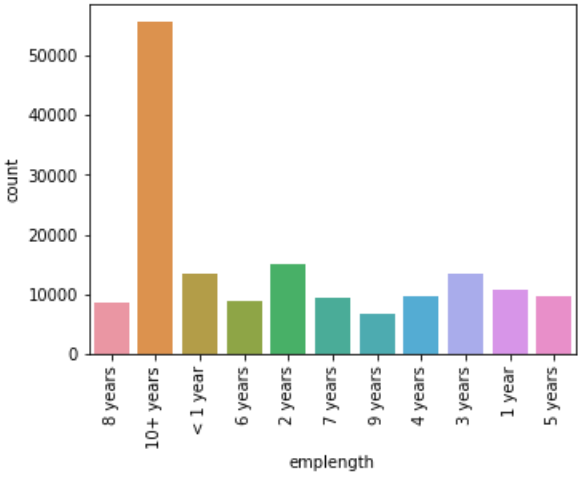





count
40000
30000
20000
10000
0
A
C
D
B
E
F
G
grade


count
120000
100000
80000
60000
40000
20000
0
36 months
60 months
term


count
80000
70000
60000
50000
40000
30000
20000
10000
0
MORTGAGE
RENT
OWN
ANY
homeownership


count
50000
40000
30000
20000
10000
0
8 years
10+ years
< 1 year
6 years
2 years
7 years
9 years
4 years
3 years
1 year
5 years
emplength

Feature Engineering

[go back to the top]

Feature engineering is conducted mainly on the categorical features. The purpose of feature engineering is either to convert categorical variables to numerical variables or extract new features that may potentially enhance model performance. There are 5 different types of categorical features, each is treated with different strategies. The table below lists each type of categorical features and their corresponding solutions.

Type	Name	Definition	Solution
Intrinsically numerical	intrate	Interest rate	Convert to numerical
	revolutil	Credit line utilization	..
	term	Term of loan (36/60)	..
Date-time	earliestcrline	Date first credit line	Extract new feature: credit history
High cardinality	emptitle	Employment title	NLP and extract new features
	desc	User comments	..
	addrzip	Zipcode	Encoding by frequency rate
	addrstate	State	..
Ordinal	grade	Loan risk grade	Label encoding
	subgrade	Loan risk subgrade	..
	emplength	Months employed	..
Other	applicationtype	Application type	Only one type. Drop the feature.
	homeownership	Home ownership	One-hot encoding (OHE)
	initialliststatus	Initial list status	..
	purpose	Purpose of loan	..
	Isincv	Is income verified?	..

Categorical Feature Treatment

Categorical features are treated one-by-one based on the nature of each feature. After the treatment of each categorical feature, that feature name is removed from the categorical variables and a new feature is added into the numerical variables.

Feature 1. "addrstate" - convert to frequency

The address state has over 48 unique values. It is hard to use dummy encoding for all of them. However, the frequency of each state may be associated with the default rate. So frequency encoding is performed on the address state feature.

In [45]:
```
addrstate_freq = pd.DataFrame({'addrstate': df_2014.addrstate.value_counts(normalize
= True).index, \
                               'state_freq': df_2014.addrstate.value_counts(normalize
 = True).values})
```

In [46]:
```
df_2014 = pd.merge(df_2014, addrstate_freq, how = 'left', on = 'addrstate')
```

In [47]:
```
cat_vars.remove('addrstate')
num_vars.append('state_freq')
```

In [48]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
17 59 (169475, 77)
```

Feature 2. 'addrzip' - convert to numeric zip and frequency.

In the raw data, the address zipcode is expressed in the format of "123XX", where the last two digits of the zipcode are masked. In this process, the first three digits of address zipcodes are extracted as numerical features by stripping "XX" from the entries. Then another feature "zip_freq" based on the frequency of zipcodes is also created.

In [49]:
```
df_2014.addrzip = df_2014.addrzip.apply(lambda x: int(x.strip('xx')))

addrzip_freq = pd.DataFrame({'addrzip': df_2014.addrzip.value_counts(normalize = True
).index, \
                             'zip_freq': df_2014.addrzip.value_counts(normalize = Tru
e).values})
```

In [50]:
```
df_2014 = pd.merge(df_2014, addrzip_freq, how = 'left', on = 'addrzip')
```

In [51]:
```
cat_vars.remove('addrzip')
num_vars.append('addrzip')
num_vars.append('zip_freq')
```

In [52]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
16 61 (169475, 78)
```

Feature 3. 'applicationtype' -remove from cat_vars

From the plot of this feature it is found that the only value of "application type" is "Individual". So this feature is dropped from the model.

In [53]:
```
# Only one value in this feature. Remove.
cat_vars.remove('applicationtype')
```

In [54]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
15 61 (169475, 78)
```

Feature 4. 'desc' - combine levels. (nan - 0, short - 1, long - 2)

This is the description that users input when applying for loans. There is only a small portion of this feature that are non-empty. The entries for description are all different. A new feature "desc_islong" is thus extracted from this column, by treating the null values as 0, those with short descriptions as 1, and those with long descriptions as 2. Natural Language Processing is used in this case to differentiate the length of decriptions.

In [55]:
```
df_2014.desc = df_2014.desc[df_2014.desc.notnull()].apply(lambda x: x.split('>')[1][:
-3] if x.find('>') != -1 else x)
```

In [56]:
```
desc_length = df_2014.desc[df_2014.desc.notnull()].apply(lambda x: len(x))
```

In [57]:
```
median = desc_length.median()
```

In [58]:
```
desc_islong = pd.DataFrame(desc_length.apply(lambda x: 1 if x < median else 2))
```

In [59]:
```
desc_islong.columns = ['desc_islong']
```

In [60]:
```
df_2014 = pd.merge(df_2014, desc_islong, how = 'left', left_index=True, right_index=True)
```

In [61]:
```
df_2014.desc_islong = df_2014.desc_islong.fillna(0).astype('int')
```

In [62]:
```
cat_vars.remove('desc')
num_vars.append('desc_islong')
```

In [63]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
14 62 (169475, 79)
```

Feature 5. 'earliestcrline' - create new feature as credit history length in days.

"Earliestcrline" refers to the date the first credit line of the user was opened, which marks the credit history of the user. It is thus more meaningful to create a new feature as "credit history" for the model. The data were first converted to the datetime format, then the time delta in days is calculated from 2017/01/01.

In [64]:
```
earliest_crl = df_2014['earliestcrline'].apply(lambda x: dt.datetime.strptime(str(x),
 '%b-%Y'))
```

In [65]:
```
# Credit history in days.
df_2014['credit_hist'] = earliest_crl.apply (lambda x: (dt.date(2018, 1, 1) - x.date
()).days)
```

In [66]:
```
cat_vars.remove('earliestcrline')
num_vars.append('credit_hist')
```

In [67]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
13 63 (169475, 80)
```

Feature 6. 'emplength' - convert to months

This feature is the length of employment. It contains ordinal levels of values from < 1 year to $\geqslant$10 years. The value of each level is extracted via Natural Language Processing and converted to months. Less than 1 year is labeled 0. For the n/a entries, it is likely that the user was unemployed, so any n/a values are replaced with a penalty term of "-999".

In [68]:
```
df_2014.emplength.unique()
```

Out[68]:
```
array(['8 years', '10+ years', '< 1 year', '6 years', '2 years', '7 years',
       '9 years', nan, '4 years', '3 years', '1 year', '5 years'], dtype=object)
```

In [69]:
```
df_2014.emplength.replace('n/a', np.nan, inplace = True)
df_2014.emplength.replace('< 1 year', '0', inplace = True)
df_2014.emplength.replace(to_replace='[^0-9]+', value = '', inplace= True, regex= Tru
e)
```

In [70]:
```
df_2014.emplength = df_2014.emplength[df_2014.emplength.notnull()].astype(int).apply(
lambda x: x * 12)
```

In [71]:
```
df_2014.emplength.fillna(-999, inplace = True)
```

In [72]:
```
cat_vars.remove('emplength')
num_vars.append('emplength')
```

In [73]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
12 64 (169475, 80)
```

Feature 7. 'emptitle' - convert to managerial (2) or non-managerial (1) or unemployed (0).

The title of employment also vary significantly so it is impossible to use dummy encoding for this feature. It can be assumed that a person with a managerial title may have better credit. As a result, the entries in this column are analyzed to find words such as "manager", "president", and so on. Those with a managerial title are labeled 1, and those without are labeled 0. The users with an n/a entry in this column are likely unemployed, so a penalty term of "-999" is also used in place of "n/a".

In [74]:
```
df_2014.emptitle = df_2014.emptitle.astype('str').apply(lambda x: x.lower())
```

In [75]:
```
df_2014.replace('nan', np.nan, inplace = True)
```

In [76]:
```
df_2014['emptype'] = np.nan
```

In [77]:
```
ismanager = df_2014.emptitle.str.contains('manager|president|officer|supervisor|direc
tor|excutive')
```

In [78]:
```
isnonmanager = (ismanager == False)
```

In [79]:
```
ismanager.fillna(False, inplace= True)
```

In [80]:
```
df_2014.emptype[ismanager] = 1
df_2014.emptype[isnonmanager] = 0
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/ind
exing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/ind
exing.html#indexing-view-versus-copy
```

In [81]:
```
df_2014.emptype.fillna(-999, inplace= True)
```

In [82]:
```
cat_vars.remove('emptitle')
num_vars.append('emptype')
```

In [83]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
11 65 (169475, 81)
```

Feature 8. 'grade' - label encoding A-G -> 1-7

The "grade" feature is pre-categorized by Lending Club based on the risk levels of loans. It is simply label-coded here from A-G to 1-7.

In [84]:
```
df_2014.grade = df_2014.grade.map({'A': 1, 'B': 2, 'C': 3, 'D': 4, 'E': 5, 'F': 6,
'G': 7})
```

In [85]:
```
cat_vars.remove('grade')
num_vars.append('grade')
```

In [86]:
```
print(len(cat_vars), len(num_vars), df_2014.shape)
```

```
10 66 (169475, 81)
```

Feature 9. 'homeownership' - leave for one-hot encoding.

The home ownership feature has only three values: mortgage, rent, or own. One-hot Encoding is performed on this feature.

In [87]:
```
df_2014.homeownership.replace('ANY', np.nan, inplace= True)
```

In [88]:
```
homeownership_OHE = pd.get_dummies(df_2014['homeownership'])
```

In [89]:
```
homeownership_OHE.columns = ['home_mort', 'home_own', 'home_rent']
```

In [90]:
```
df_2014 = pd.merge(df_2014, homeownership_OHE, how= 'left', left_index = True, right_
index= True)
```

In [91]:
```
OHE_vars = []
OHE_vars = OHE_vars + list(homeownership_OHE.columns)
```

In [92]:
```
cat_vars.remove('homeownership')
```

In [93]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
9 66 3 (169475, 84)
```

Feature 10. 'initialliststatus' - map: W: 0, F: 1.

Only two values are in this feature: 0 or 1 for W or F.

In [94]:
```
# Upper case to match the current loans data.
df_2014.initialliststatus = df_2014.initialliststatus.apply(lambda x: x.upper())
```

In [95]:
```
df_2014.initialliststatus = df_2014.initialliststatus.map({'W': 0, 'F': 1})
```

In [96]:
```
cat_vars.remove('initialliststatus')
num_vars.append('initialliststatus')
```

In [97]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
8 67 3 (169475, 84)
```

Feature 11. 'intrate' - change to float

The feature of interest rate is intrinscally numerical. It is converted to a float number by removing the "%" sign.

In [98]:
```
df_2014['intrate'] = df_2014.intrate.astype('str').transform(lambda x: float(x.strip(
'%'))/100)
```

In [99]:
```
cat_vars.remove('intrate')
num_vars.append('intrate')
```

In [100]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
7 68 3 (169475, 84)
```

Feature 12. 'isincv' - one hot encoding.

Only three different values are in this feature. It is noted that the historical entries may be in lower or upper cases but the entries in the current data have only upper cases. So they are uniformed to upper cases and One-hot Encoded.

In [101]:
```
df_2014.isincv.replace(['Not Verified', 'Source Verified', 'Verified'], ['NOT_VERIFIE
D', 'SOURCE_VERIFIED', 'VERIFIED'],\
                           inplace= True)
```

In [102]:
```
isincv_OHE = pd.get_dummies(df_2014['isincv'])
```

In [103]:
```
isincv_OHE.columns = ['isincv_nv', 'isinicv_sv', 'isincv_v']
```

In [104]:
```
df_2014 = pd.merge(df_2014, isincv_OHE, how='left', left_index = True, right_index= T
rue)
```

In [105]:
```
OHE_vars = OHE_vars + list(isincv_OHE.columns)
```

In [106]:
```
cat_vars.remove('isincv')
```

In [107]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
6 68 6 (169475, 87)
```

Feature 13. 'purpose' - one-hot encoding.

This feature describes the purpose of loan application. It is One-hot Encoded.

In [108]:
```
purpose_OHE = pd.get_dummies(df_2014.purpose)
```

In [109]:
```
purpose_OHE.columns = ['pur_' + var for var in purpose_OHE.columns.values]
```

In [110]:
```
df_2014 = pd.merge (df_2014, purpose_OHE, how= 'left', left_index= True, right_index=
 True)
```

In [111]:
```
OHE_vars =  OHE_vars + list(purpose_OHE.columns)
```

In [112]:
```
cat_vars.remove('purpose')
```

In [113]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
5 68 19 (169475, 100)
```

Feature 14. 'revolutil' - convert to float

This feature indicates the utilization rate of users' revolving credit limit. It is intrinsically numeric with removal of the "%" sign.

In [114]:
```
df_2014.revolutil = df_2014.revolutil.astype('str').transform(lambda x: float(x.strip
('%'))/100)
```

In [115]:
```
cat_vars.remove('revolutil')
num_vars.append('revolutil')
```

In [116]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
4 69 19 (169475, 100)
```

Feature 15. 'subgrade' - label encoding

Similar to the grade feature, this feature is ordinal and coded by label-coding.

In [117]:
```
subgrd = df_2014.subgrade.sort_values().unique()
```

In [118]:
```
df_2014.subgrade = df_2014.subgrade.map({subgrd[i]: i + 1 for i in range(len(subgrd
))})
```

In [119]:
```
cat_vars.remove('subgrade')
num_vars.append('subgrade')
```

In [120]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
3 70 19 (169475, 100)
```

Feature 16. 'term' - remove 'months'

Only two types are in this feature, 36 months or 60 months. They are converted to numbers.

In [121]:
```
df_2014.term = df_2014.term.apply(lambda x: ''.join(d for d in x if d.isdigit()))
```

In [122]:
```
df_2014.term = df_2014.term.astype(int)
```

In [123]:
```
cat_vars.remove('term')
num_vars.append('term')
```

In [124]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
2 71 19 (169475, 100)
```

In [125]:
```
df_2014.applicationtype.unique()
```

Out[125]:
```
array(['INDIVIDUAL'], dtype=object)
```

Feature 17. 'issued' - convert to date - not in training set.

This feature only marks the issued date of each loan, which is useful for train-test splitting, but not used in the training model.

In [126]:
```
df_2014.issued = df_2014.issued.astype('str').apply(lambda x: dt.datetime.strptime(st
r(x), '%b-%Y'))
```

In [127]:
```
cat_vars.remove('issued')
```

Feature 18. 'loanstatus' - Target - 'charged_off' = 1, 'fully_paid' = 0

This is the target feature. 1 for "charged off", 0 for "fully paid".

In [128]:
```
df_2014.loanstatus = df_2014.loanstatus.map({'Charged Off': 1, 'Default': 1, 'Fully P
aid': 0})
```

In [129]:
```
cat_vars.remove('loanstatus')
```

In [130]:
```
print(len(cat_vars), len(num_vars), len(OHE_vars), df_2014.shape)
```

```
0 71 19 (169475, 100)
```

Feature Responses Visualization

In this section, several important features are plotted with their responses of loan status. The values of each feature are assigned into different bins and the average default rates (mean of loan status) in each bin are plotted using bar plot.

dti - Debt to Income Ratio

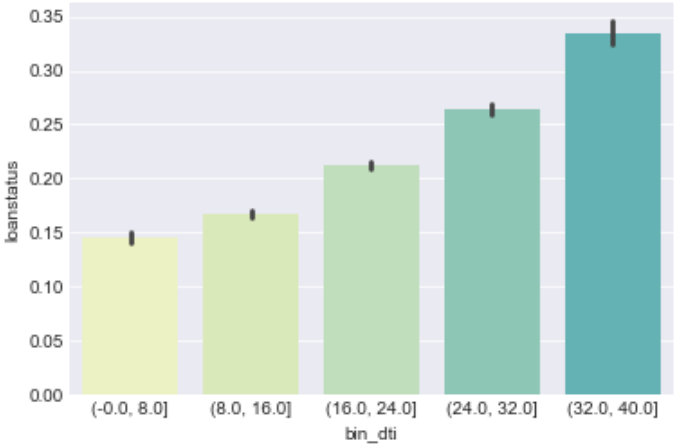
In [131]:
```
%matplotlib inline
sns.set_style('darkgrid')
```

In [132]:
```
df_2014['bin_dti'] = pd.cut(df_2014.dti, bins=5, precision=0)
```

In [133]:
```
sns.barplot(x='bin_dti', y= 'loanstatus', data = df_2014, palette=sns.color_palette(
"YlGnBu", 10))
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[133]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0d0909e8>



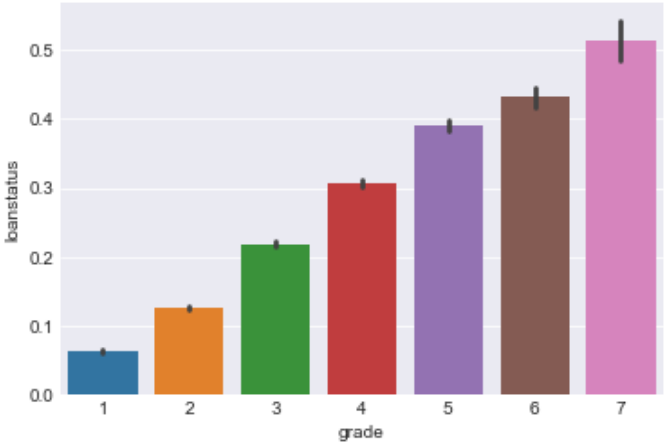
The figure above shows that the debt-to-income ratio is very related with the loan status. Higher dti leads to higher chance of default.

grade and subgrade

In [134]:
```
sns.barplot(x = 'grade', y = 'loanstatus', data = df_2014)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

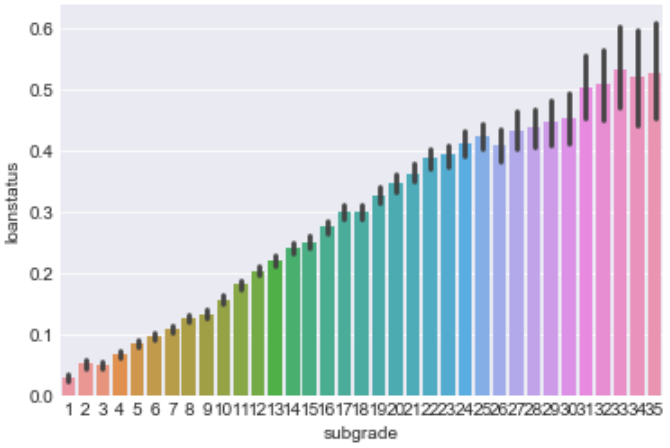
Out[134]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0d346438>



In [135]:
```
sns.barplot(x = 'subgrade', y = 'loanstatus', data = df_2014)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[135]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0d838ba8>



Grade and subgrade clearly show an increasing trend with default rate. They are good indicators of risk levels.

annualinc - Annual Income

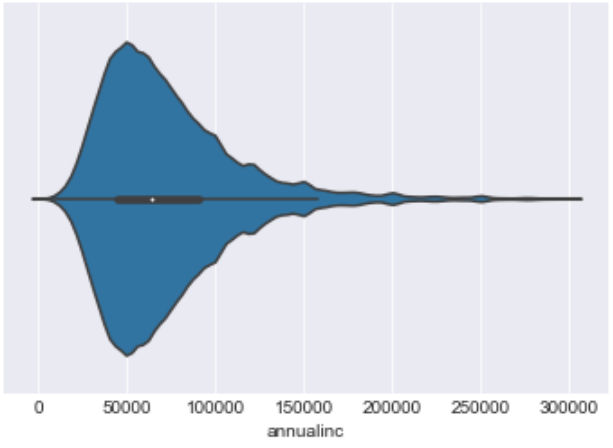
The distribution of annual income is very skewed based on the violin plot, so the upper range of annual income is cut from 200000, with bin size of 25000.

In [136]:
```
df_2014['bin_annualinc'] = pd.cut(df_2014.annualinc, bins=[0, 25000, 50000, 75000, \
                                                           100000, 125000, 150000, \
                                                           175000, 200000, max(df_201
4.annualinc)], precision=0)
```

In [137]:
```
sns.violinplot(x='annualinc', data= df_2014[df_2014['annualinc'] <300000])
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:598: FutureWarning: rem
ove_na is deprecated and is a private function. Do not use.
  kde_data = remove_na(group_data)
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:826: FutureWarning: rem
ove_na is deprecated and is a private function. Do not use.
  violin_data = remove_na(group_data)
```

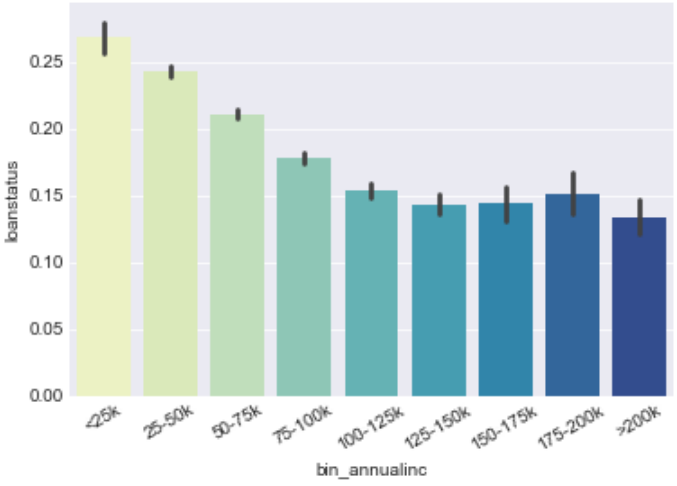
Out[137]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1ba85e80>



In [138]:
```
fig = sns.barplot(x='bin_annualinc', y= 'loanstatus', data = df_2014, palette=sns.color_palette("YlGnBu", 10))
fig.set_xticklabels(labels=['<25k', '25-50k', '50-75k', '75-100k', '100-125k', '125-150k',\
                            '150-175k', '175-200k', '>200k'], rotation = 30)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[138]:
```
[Text(0,0,'<25k'),
 Text(0,0,'25-50k'),
 Text(0,0,'50-75k'),
 Text(0,0,'75-100k'),
 Text(0,0,'100-125k'),
 Text(0,0,'125-150k'),
 Text(0,0,'150-175k'),
 Text(0,0,'175-200k'),
 Text(0,0,'>200k')]
```



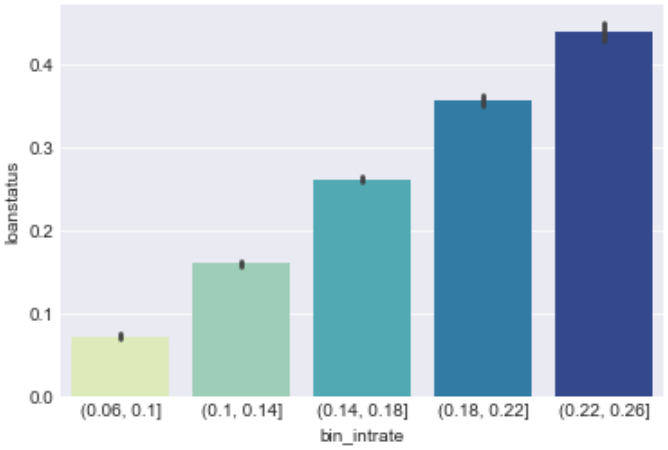
The fitures show that the higher annual income, the lower chance of default.

Intrate - Interest Rate

In [139]:
```
df_2014['bin_intrate'] = pd.cut(df_2014.intrate, bins=5, precision=2)
```

In [140]:
```
fig = sns.barplot(x='bin_intrate', y= 'loanstatus', data = df_2014, palette=sns.color
_palette("YlGnBu", 5))
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```



The Interest Rate clearly shows a positive linear relationship with default rate.

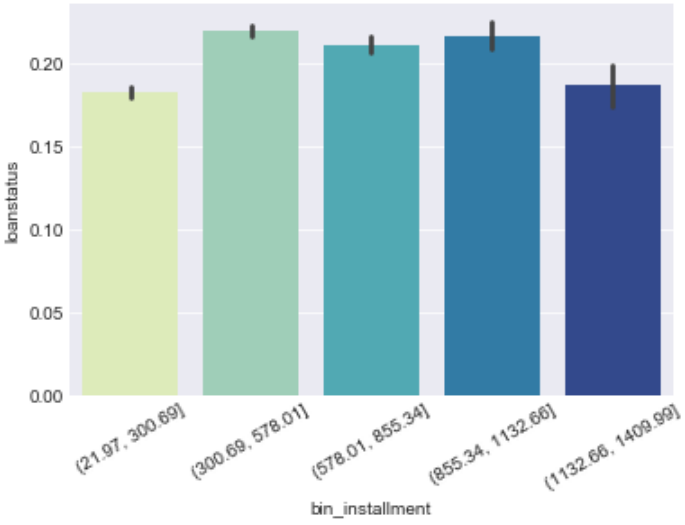
Installment

In [141]:
```
df_2014['bin_installment'] = pd.cut(df_2014.installment, bins=5, precision=2)
```

In [142]:
```
fig = sns.barplot(x='bin_installment', y= 'loanstatus', data = df_2014, palette=sns.c
olor_palette("YlGnBu", 5))
plt.xticks(rotation = 30)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[142]:
```
(array([0, 1, 2, 3, 4]), <a list of 5 Text xticklabel objects>)
```



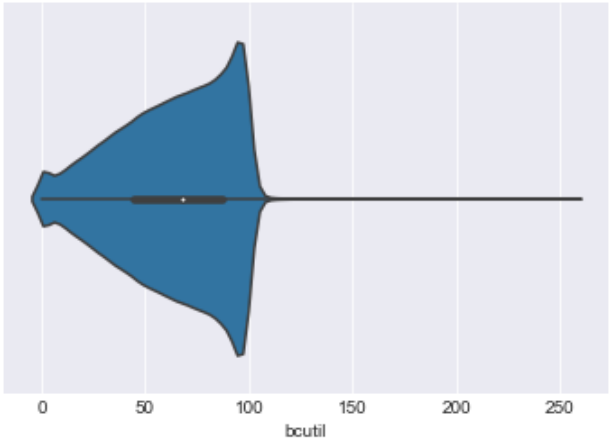
The installment amount may have a parabolic relationship with the default rate.

bcutil - Ration of total current loan balance to bank card credit limit (Portion of Balance)

In [143]:
```
sns.violinplot(x='bcutil', data= df_2014)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:598: FutureWarning: rem
ove_na is deprecated and is a private function. Do not use.
  kde_data = remove_na(group_data)
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:826: FutureWarning: rem
ove_na is deprecated and is a private function. Do not use.
  violin_data = remove_na(group_data)
```

Out[143]: <matplotlib.axes._subplots.AxesSubplot at 0x1a109598d0>



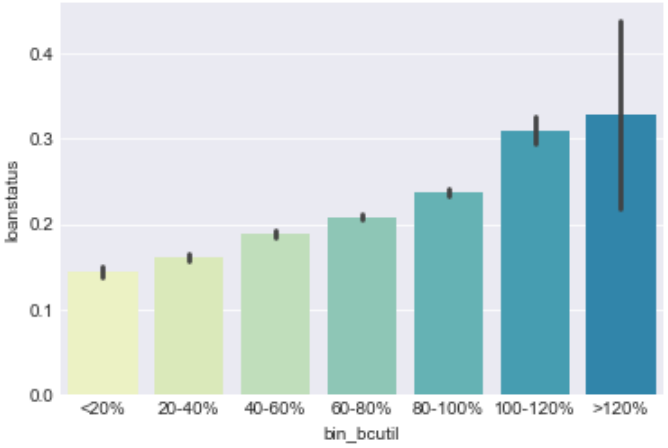
The distribution of the Portion of Balance is also very skewed. A interesting finding from this figure is that most of the borrowers on Lending Clube have used almost > 90% of their credit limit. So Lending Club may be dealing with highly risky borrowers that may not be able to borrow from banks. There are a small amount of borrowers with loan balance > 3 times of credit limit. These may mark even higher risk of default.

In [144]:
```
df_2014['bin_bcutil'] = pd.cut(df_2014.bcutil, bins=[0, 20, 40, 60, \
                                                     80, 100, 120, \
                                                     max(df_2014.bcutil)], prec
ision=0)
```

In [145]:
```
fig = sns.barplot(x='bin_bcutil', y= 'loanstatus', data = df_2014, palette=sns.color_
palette("YlGnBu", 10))
fig.set_xticklabels(labels=['<20%', '20-40%', '40-60%', '60-80%', '80-100%', '100-12
0%', '>120%'])
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: re
move_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[145]:
```
[Text(0,0,'<20%'),
 Text(0,0,'20-40%'),
 Text(0,0,'40-60%'),
 Text(0,0,'60-80%'),
 Text(0,0,'80-100%'),
 Text(0,0,'100-120%'),
 Text(0,0,'>120%')]
```



It is not surprising to see an increasing trend of default rate vs. Portion of Balance. The variance of default rate also increase with higher Portion of Balance.

Credit History

In [146]:
```
df_2014['bin_credit_hist'] = pd.cut(df_2014.credit_hist, bins=5, precision=0)
```

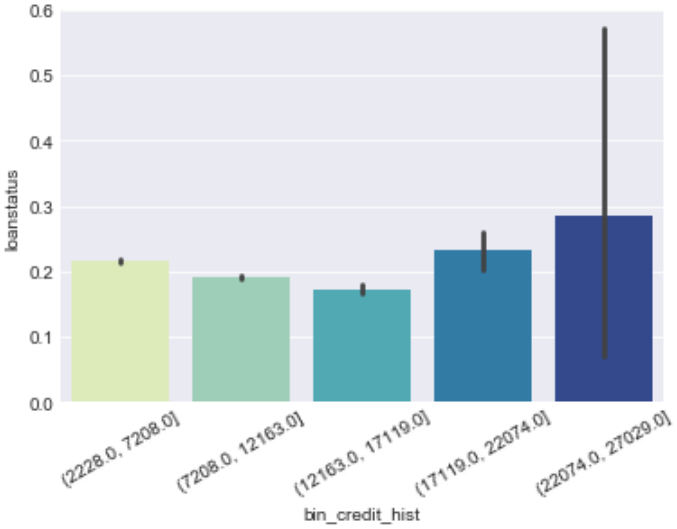
In [147]:
```
df_2014.bin_credit_hist.value_counts()
```

Out[147]:
```
(2228.0, 7208.0]      97456
(7208.0, 12163.0]     62096
(12163.0, 17119.0]     9150
(17119.0, 22074.0]      759
(22074.0, 27029.0]       14
Name: bin_credit_hist, dtype: int64
```

In [148]:
```
fig = sns.barplot(x='bin_credit_hist', y= 'loanstatus', data = df_2014, palette=sns.color_palette("YlGnBu", 5))
plt.xticks(rotation = 30)
```

```
/anaconda3/lib/python3.6/site-packages/seaborn/categorical.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

Out[148]: (array([0, 1, 2, 3, 4]), <a list of 5 Text xticklabel objects>)



No apparent trend is seen between credit history and default rate, although the variation seems to increase with longer credit history.

XGBoost Model Building

[go back to the top]

Train/Test Splitting

Because this model uses historic data to predict future performances, when spliting the train/test datasets, the historic 2014 loan data are splitted into two parts chronologically: Dec/Nov/Oct data as the test set, and Jan - Sep as the train set.

In [149]:
```
num_vars.remove('id')
```

In [150]:
```
train_vars = num_vars + OHE_vars
```

In [151]:
```
df_train = df_2014[train_vars + ['loanstatus']][df_2014.issued < dt.date(2014, 10, 1)]
df_test = df_2014[train_vars + ['loanstatus']][df_2014.issued >= dt.date(2014, 10, 1)]
```

In [152]:
```
df_train.shape
```

Out[152]: (127364, 90)

In [153]:
```
df_test.shape
```

Out[153]: (42111, 90)

Parameter Auto-tuning

In this section both packages of XGBoost and BayesianOptimization are used. XGBoost is the model buiding algorithm. There are a number of parameters used in XGBoost, of which 5 parameters are of greater importance and given fine tuning. The parameter tuning is conducted with the Bayesian Optimization algorithm.

Bayesian Optmization is a constrained global optimization package built upon bayesian inference and gaussian process. It attempts to find the maximum value of an unknown function based on a few starting points, in as few iterations as possible. This technique is particularly suited for optimization of high cost functions. In this case the target function is the cross-validation score from XGBoost models. The function is defined as "xgb_evaluate", where the output of this function is the best cross-validation score for each set of parameters. The cross-validation uses "auc" (area under the ROC curve) as the scoring mectrics. The early stop callback is set at 50, meaning that if no improvement of score after 50 interations, the procedure is stopped.

The Bayesian Optimization process then maximizes the target function based on 8 initial points and interate 40 times to yield the best set of parameters leading to the highst "auc" scores.

In [154]:
```
import xgboost as xgb
from bayes_opt import BayesianOptimization
from sklearn import metrics, preprocessing, cross_validation
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/cross_validation.py:41: DeprecationWar
ning: This module was deprecated in version 0.18 in favor of the model_selection modu
le into which all the refactored classes and functions are moved. Also note that the
interface of the new CV iterators are different from that of this module. This module
will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [155]:
```
x_train = df_train[train_vars]
y_train = df_train['loanstatus']
x_test = df_test[train_vars]
y_test = df_test['loanstatus']
```

In [156]:
```
dtrain = xgb.DMatrix(x_train, y_train, missing= np.nan)
dtest = xgb.DMatrix(x_test, missing= np.nan)
```

In [157]:
```
def xgb_evaluate(min_child_weight,
                 colsample_bytree,
                 max_depth,
                 subsample,
                 gamma):
    params = dict()
    params['objective'] = 'binary:logistic'
    params['eta'] = 0.1
    params['max_depth'] = int(max_depth)
    params['min_child_weight'] = int(min_child_weight)
    params['colsample_bytree'] = colsample_bytree
    params['subsample'] = subsample
    params['gamma'] = gamma
    params['verbose_eval'] = True


    cv_result = xgb.cv(params, dtrain,
                       num_boost_round=15000,
                       nfold=5,
                       metrics={'auc'},
                       seed=1234,
                       stratified=True,
                       callbacks=[xgb.callback.early_stop(50)])

    return cv_result['test-auc-mean'].max()
```

In [158]:
```
xgb_BO = BayesianOptimization(xgb_evaluate,
                              {'max_depth': (3, 10),
                               'min_child_weight': (0, 100),
                               'colsample_bytree': (0.5, 1),
                               'subsample': (0.5, 1),
                               'gamma': (0, 2)
                              }
                             )
```

In [159]:
```
%%time
xgb_BO.maximize(init_points=8, n_iter=40)
```

```
Initialization
-------------------------------------------------------------------------------------
--------------------------
 Step |   Time |      Value |   colsample_bytree |     gamma |   max_depth |   min_ch
ild_weight |   subsample |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[87]    train-auc:0.787904+0.0016154    test-auc:0.728826+0.00298646

    1 | 05m12s |    0.72883 |             0.7858 |    1.3136 |      9.8255 |
   74.5920 |      0.6287 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[127]   train-auc:0.770267+0.000695463  test-auc:0.730134+0.00295456

    2 | 03m32s |    0.73013 |             0.5611 |    0.3318 |      6.8941 |
   67.6846 |      0.5440 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[264]   train-auc:0.782409+0.00118676   test-auc:0.730636+0.00238837

    3 | 05m46s |    0.73064 |             0.5916 |    1.8618 |      5.1459 |
   80.7077 |      0.7117 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[295]   train-auc:0.777957+0.000757256  test-auc:0.730982+0.00303475

    4 | 06m03s |    0.73098 |             0.7135 |    1.7018 |      4.3086 |
   15.5271 |      0.6822 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[77]    train-auc:0.79888+0.00105672    test-auc:0.727114+0.00294511

    5 | 03m58s |    0.72711 |             0.5695 |    0.3479 |      9.5931 |
   43.2013 |      0.6223 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[493]   train-auc:0.770101+0.000563423  test-auc:0.731915+0.0022128

    6 | 09m54s |    0.73192 |             0.9171 |    1.5230 |      3.2222 |
   20.6057 |      0.8540 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[117]   train-auc:0.788506+0.00129176   test-auc:0.729472+0.00321068

    7 | 04m50s |    0.72947 |             0.7991 |    1.5887 |      6.8414 |
   23.6638 |      0.7984 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[98]    train-auc:0.797058+0.000929575  test-auc:0.728539+0.00353873

    8 | 06m33s |    0.72854 |             0.9804 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.36693907
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 59, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0003428
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 3, 'warnflag':
2}
  " state: %s" % convergence_dict)

   1.2263 |      8.5930 |           60.2247 |      0.7299 |
Bayesian Optimization
--------------------------------------------------------------------------------
--------------------------
 Step |   Time |      Value |   colsample_bytree |     gamma |   max_depth |   min_ch
ild_weight |   subsample |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[475]   train-auc:0.763952+0.000411999  test-auc:0.732037+0.0024874

    9 | 10m01s |    0.73204 |            0.9978 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.00555044
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 52, 'nit': 4, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

   0.2559 |      3.5493 |           99.9485 |      0.7863 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[273]   train-auc:0.782108+0.000787822  test-auc:0.729696+0.00339703

   10 | 09m12s |    0.72970 |            0.9813 |    0.0049 |      4.8693 |
    0.2297 |      0.9932 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:335: UserWarni
ng: Predicted variances smaller than 0. Setting those variances to 0.
  warnings.warn("Predicted variances smaller than 0. "

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[404]   train-auc:0.75902+0.000433618   test-auc:0.731714+0.00248016

   11 | 09m14s |    0.73171 |            0.9831 |    0.0452 |      3.2714 |
   86.7043 |      0.5200 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[512]   train-auc:0.764524+0.000585039  test-auc:0.73204+0.00260216

   12 | 08m21s |    0.73204 |            0.5431 |    0.1201 |      3.1949 |
   82.3529 |      0.9906 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001757
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 50, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.17972608
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0003847
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[419]   train-auc:0.765459+0.000716954  test-auc:0.7316+0.00280544

   13 | 09m01s |    0.73160 |             0.7980 |    0.0785 |      3.0846 |
   16.8290 |      0.9043 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.09533139
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 56, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[137]   train-auc:0.799951+0.00115173   test-auc:0.729901+0.00270172

   14 | 06m03s |    0.72990 |             0.5138 |    0.0795 |      8.3378 |
   99.6047 |      0.9672 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  2.84777152
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[443]   train-auc:0.759977+0.000608692  test-auc:0.731459+0.00279842

   15 | 06m41s |    0.73146 |             0.5102 |    0.0199 |      3.0055 |
   97.2342 |      0.6625 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[495]   train-auc:0.765929+0.000463688  test-auc:0.732224+0.00244763

   16 | 12m07s |    0.73222 |             0.9788 |    0.2154 |      3.0796 |
   71.2835 |      0.9668 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -9.10613599
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 52, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  4.58098075
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 55, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[347]   train-auc:0.761901+0.000560147  test-auc:0.731026+0.0029354

   17 | 06m20s |    0.73103 |             0.6036 |    1.9156 |      3.0251 |
    0.6868 |      0.9258 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[408]   train-auc:0.760494+0.000818102  test-auc:0.731522+0.00287172

   18 | 06m34s |    0.73152 |             0.5696 |    0.8966 |      3.0008 |
   33.0816 |      0.5580 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[579]   train-auc:0.769406+0.000300602  test-auc:0.732133+0.00231025

   19 | 13m14s |    0.73213 |             0.9110 |    0.2467 |      3.5537 |
   77.3981 |      0.9706 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[422]   train-auc:0.761704+0.000634356  test-auc:0.731863+0.00266652

   20 | 09m46s |    0.73186 |             0.9104 |    0.3241 |      3.3780 |
   78.8026 |      0.8361 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0001017
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0043718
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 52, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[350]   train-auc:0.756883+0.000683001  test-auc:0.731821+0.00291028

   21 | 08m24s |    0.73182 |             0.8794 |    1.8964 |      3.2264 |
   93.1923 |      0.9890 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.000370
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 82, 'nit': 9, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -7.21961260
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 71, 'nit': 9, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0010305
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 69, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[479]   train-auc:0.765581+0.000514553  test-auc:0.731474+0.00247187

   22 | 10m51s |    0.73147 |             0.8673 |    0.1433 |      3.1972 |
   45.8883 |      0.9924 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0044927
9]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -8.73992103
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 72, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -3.90151690
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 56, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[429]   train-auc:0.763065+0.000408139  test-auc:0.731457+0.00290191

   23 | 08m03s |    0.73146 |             0.6521 |    1.7805 |      3.1887 |
   33.0244 |      0.9947 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  8.13454390
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 65, 'nit': 9, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  6.31511211
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[397]   train-auc:0.759584+0.000776005  test-auc:0.732013+0.0025079

   24 | 09m51s |    0.73201 |             0.9797 |    0.0000 |      3.4534 |
   95.5402 |      1.0000 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0003062
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 86, 'nit': 9, 'warnflag':
2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[534]   train-auc:0.771278+0.000782231  test-auc:0.731596+0.00279661

   25 | 12m09s |    0.73160 |            0.9285 |    0.0447 |      3.1035 |
   27.7662 |      0.8856 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0119785
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 63, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.005186
2]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 67, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001428
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 50, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -3.10689211
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 8, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0092519
5]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 54, 'nit': 3, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.24275684
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[528]   train-auc:0.766703+0.000807204  test-auc:0.731985+0.00251665

   26 | 09m51s |    0.73199 |            0.6862 |    0.0604 |      3.1593 |
   73.6900 |      0.9239 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001558
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 62, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.000600
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 3, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0046488
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 61, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.000286
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 72, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[339]   train-auc:0.760691+0.000708074  test-auc:0.730556+0.0027305

   27 | 05m30s |    0.73056 |             0.5696 |    0.1263 |      3.0003 |
    0.3136 |      0.5102 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0002031
6]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 54, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.42604113
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 61, 'nit': 8, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -5.45531511
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  2.57939100
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 54, 'nit': 3, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0041358
6]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 61, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.26199627
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 84, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[484]   train-auc:0.764083+0.000786323  test-auc:0.731994+0.00250623

   28 | 09m52s |    0.73199 |             0.7863 |    0.1254 |      3.0996 |
   84.7682 |      0.9475 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001242
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 73, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001572
5]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 66, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.41813250
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 48, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0003325
4]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.67759135
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 70, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.003539
4]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 60, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.27829441
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[541]   train-auc:0.768878+0.000417256  test-auc:0.732392+0.00245547

   29 | 12m41s |    0.73239 |             0.9668 |    0.0804 |      3.7150 |
   60.8399 |      0.9358 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0016799
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 47, 'nit': 3, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0022854
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 55, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[136]   train-auc:0.794395+0.00112552   test-auc:0.729856+0.0023695

   30 | 06m50s |    0.72986 |             0.7011 |    0.0773 |      7.0201 |
   82.1718 |      0.9744 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0025456
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 65, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0006780
2]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 52, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  1.77682959
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 8, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[51]    train-auc:0.883727+0.00128184   test-auc:0.724439+0.00333911

   31 | 04m08s |    0.72444 |             0.5077 |    1.9193 |      9.7440 |
    0.2684 |      0.9959 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[362]   train-auc:0.763588+0.000614515  test-auc:0.731464+0.00325041

   32 | 08m24s |    0.73146 |             0.9640 |    1.0545 |      3.0378 |
    6.7360 |      0.6267 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -9.18542310
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 73, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0015832
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 61, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[429]   train-auc:0.761006+0.000476397  test-auc:0.731894+0.0027248

   33 | 06m52s |    0.73189 |             0.5266 |    1.8507 |      3.1247 |
   62.6717 |      0.9930 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0045280
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 64, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  4.52548265
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 66, 'nit': 4, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.001121
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0066150
4]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 51, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0005982
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 66, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[201]   train-auc:0.771791+0.000536753  test-auc:0.730072+0.00222458

   34 | 08m03s |    0.73007 |             0.9792 |    1.9814 |      5.5677 |
   99.5109 |      0.6055 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0034613
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 68, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[493]   train-auc:0.767867+0.000594361  test-auc:0.732074+0.00244028

   35 | 10m42s |    0.73207 |             0.8776 |    0.1947 |      3.0370 |
   37.4982 |      0.8586 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([  5.04851341
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 47, 'nit': 3, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.09435821
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 76, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0005482
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 56, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[357]   train-auc:0.764056+0.00058729   test-auc:0.730404+0.0024271

   36 | 08m02s |    0.73040 |             0.9435 |    1.9364 |      3.6933 |
    0.0353 |      0.5775 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0026781
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 65, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0011934
4]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 57, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[507]   train-auc:0.764513+0.000643588  test-auc:0.732281+0.00292132

   37 | 07m37s |    0.73228 |             0.5012 |    0.0496 |      3.2539 |
   65.9431 |      0.9829 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.42976642
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 56, 'nit': 6, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[538]   train-auc:0.766364+0.000774715  test-auc:0.731947+0.00242614

   38 | 10m57s |    0.73195 |             0.8048 |    0.0392 |      3.3632 |
   89.0584 |      0.9943 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0035941
5]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 63, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0004149
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 59, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.51280144
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 55, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0003591
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 59, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001840
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 74, 'nit': 6, 'warnflag':
2}
  " state: %s" % convergence_dict)
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[453]   train-auc:0.76465+0.000790475   test-auc:0.732132+0.0024415

   39 | 10m46s |    0.73213 |             0.9985 |    0.4488 |      3.2895 |
   62.7708 |      0.8729 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0028717
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 55, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.05785036
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 59, 'nit': 8, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0171276
9]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 54, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -2.58535147
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 68, 'nit': 7, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[421]   train-auc:0.762854+0.000434714  test-auc:0.732073+0.00274008

   40 | 09m11s |    0.73207 |             0.8781 |    0.0416 |      3.0927 |
   55.8416 |      0.8445 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -7.39673596
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 55, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.41761375
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ -1.55765338
e-05]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 53, 'nit': 5, 'warnfla
g': 2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[553]   train-auc:0.768022+0.000463482  test-auc:0.73186+0.00248834

   41 | 09m49s |    0.73186 |             0.6725 |    0.1274 |      3.2319 |
   60.6612 |      0.9870 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0019160
4]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 66, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0004911
9]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 58, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[224]   train-auc:0.78579+0.00104717    test-auc:0.730078+0.00260314

   42 | 09m29s |    0.73008 |             0.9935 |    1.9106 |      5.3480 |
   50.0230 |      0.9723 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[147]   train-auc:0.796125+0.000969468  test-auc:0.729182+0.00320906

   43 | 08m09s |    0.72918 |             0.9969 |    0.0409 |      6.6822 |
   33.3649 |      0.9417 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0001388
7]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 62, 'nit': 8, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.003844
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 62, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[68]    train-auc:0.846031+0.000699342  test-auc:0.724962+0.00261555

   44 | 06m34s |    0.72496 |             0.9035 |    0.0871 |      9.9590 |
   13.1619 |      0.7603 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0002576
5]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 54, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[257]   train-auc:0.768009+0.00084056   test-auc:0.731126+0.00284516

   45 | 08m20s |    0.73113 |             0.9497 |    1.9554 |      4.6478 |
   67.2059 |      0.9848 |
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0004350
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 49, 'nit': 2, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0033916
6]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 73, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[447]   train-auc:0.765528+0.00040792   test-auc:0.732275+0.00279056

   46 | 10m16s |    0.73227 |             0.9970 |    1.9748 |      3.0046 |
   40.3501 |      0.7354 |
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[157]   train-auc:0.79138+0.00115238    test-auc:0.729664+0.00280144

   47 | 08m25s |    0.72966 |             0.9778 |    0.0352 |      6.7052 |
   55.3277 |      0.9521 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0001171
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 57, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.000604
3]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 60, 'nit': 5, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0002121
5]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 61, 'nit': 7, 'warnflag':
2}
  " state: %s" % convergence_dict)
/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([-0.0015403
1]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 57, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)

Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 50 rounds.
Stopping. Best iteration:
[447]   train-auc:0.764768+0.000757547  test-auc:0.732107+0.00233837

   48 | 10m30s |    0.73211 |             0.9932 |    1.9361 |      3.1202 |
   55.9112 |      0.8580 |

/anaconda3/lib/python3.6/site-packages/sklearn/gaussian_process/gpr.py:457: UserWarni
ng: fmin_l_bfgs_b terminated abnormally with the  state: {'grad': array([ 0.0001658
8]), 'task': b'ABNORMAL_TERMINATION_IN_LNSRCH', 'funcalls': 52, 'nit': 4, 'warnflag':
2}
  " state: %s" % convergence_dict)

CPU times: user 23h 50min 24s, sys: 14min 39s, total: 1d 5min 4s
Wall time: 6h 43min 28s
```

In [160]:
```
BO_scores = pd.DataFrame(xgb_BO.res['all']['params'])
BO_scores['score'] = pd.DataFrame(xgb_BO.res['all']['values'])
BO_scores = BO_scores.sort_values(by='score',ascending=False).reset_index()
BO_scores.head()
```

Out[160]:

	index	colsample_bytree	gamma	max_depth	min_child_weight	subsample	score
0	20	0.966764	0.080371	3.714995	60.839938	0.935822	0.732392
1	28	0.501193	0.049590	3.253927	65.943135	0.982906	0.732281
2	37	0.997029	1.974814	3.004641	40.350106	0.735441	0.732275
3	7	0.978806	0.215396	3.079573	71.283494	0.966837	0.732224
4	10	0.911004	0.246685	3.553736	77.398079	0.970618	0.732133

Find Best Parameters

The resulted parameters are sorted in a descending order of their corresponding scores. The best parameters yield an AUC score of 0.733219. The best parameters are then input to the cross-validation function to find the best interation.

- 'max_depth': 3,
- 'min_child_weight': 61,
- 'colsample_bytree': 0.5123,
- 'subsample': 0.8830,
- 'gamma': 1.9438,
- 'objective': 'binary:logistic',
- 'eta': 0.02,
- 'seed': 1234

In [161]:
```
best_params = dict()

best_params['max_depth'] = int(BO_scores['max_depth'][0])
best_params['min_child_weight'] = int(BO_scores['min_child_weight'][0])
best_params['colsample_bytree'] = BO_scores['colsample_bytree'][0]
best_params['subsample'] = BO_scores['subsample'][0]
best_params['gamma'] = BO_scores['gamma'][0]

best_params['objective'] = 'binary:logistic'
best_params['eta'] = 0.02
best_params['seed'] = 1234
```

In [162]:
```
best_params
```

Out[162]:
```
{'colsample_bytree': 0.96676375041398677,
 'eta': 0.02,
 'gamma': 0.080370624378502509,
 'max_depth': 3,
 'min_child_weight': 60,
 'objective': 'binary:logistic',
 'seed': 1234,
 'subsample': 0.93582224874989284}
```

In [163]:
```
cv_result = xgb.cv(best_params, dtrain,
                   num_boost_round=15000,
                   nfold=5,
                   metrics={'auc'},
                   seed=1234,
                   stratified=True,
                   callbacks=[xgb.callback.early_stop(200)],
                   verbose_eval=50)


best_iteration = len(cv_result)
best_score = cv_result['test-auc-mean'].max()
print("Best score %f, best iteration %d" % (best_score,best_iteration) )
```

```
Multiple eval metrics have been passed: 'test-auc' will be used for early stopping.

Will train until test-auc hasn't improved in 200 rounds.
[0]     train-auc:0.694535+0.0014457    test-auc:0.692682+0.00308518
[50]    train-auc:0.70876+0.00102308    test-auc:0.706725+0.00368996
[100]   train-auc:0.712997+0.00103017   test-auc:0.710229+0.00370518
[150]   train-auc:0.718559+0.000892052  test-auc:0.714871+0.00381422
[200]   train-auc:0.723451+0.000804958  test-auc:0.718569+0.00373243
[250]   train-auc:0.727171+0.000783027  test-auc:0.721269+0.00357146
[300]   train-auc:0.730044+0.000714466  test-auc:0.723092+0.00351995
[350]   train-auc:0.732334+0.000753368  test-auc:0.724428+0.00339445
[400]   train-auc:0.734198+0.000708377  test-auc:0.725414+0.00339283
[450]   train-auc:0.735796+0.000692314  test-auc:0.72616+0.00331707
[500]   train-auc:0.737203+0.000608134  test-auc:0.726805+0.00328542
[550]   train-auc:0.738492+0.000596256  test-auc:0.72738+0.00323954
[600]   train-auc:0.739674+0.000560202  test-auc:0.72781+0.00321021
[650]   train-auc:0.740822+0.00055498   test-auc:0.728257+0.00315154
[700]   train-auc:0.741953+0.000559091  test-auc:0.728692+0.00311818
[750]   train-auc:0.743048+0.000581125  test-auc:0.729069+0.00307439
[800]   train-auc:0.744116+0.000606372  test-auc:0.729443+0.00302587
[850]   train-auc:0.745131+0.00057337   test-auc:0.729729+0.00297337
[900]   train-auc:0.746111+0.000585077  test-auc:0.730041+0.00295228
[950]   train-auc:0.747036+0.000602919  test-auc:0.730274+0.00290823
[1000]  train-auc:0.747954+0.000610052  test-auc:0.730497+0.00288353
[1050]  train-auc:0.748808+0.000610566  test-auc:0.730661+0.00282875
[1100]  train-auc:0.749617+0.000619639  test-auc:0.730849+0.00281615
[1150]  train-auc:0.750436+0.000609648  test-auc:0.731075+0.00282007
[1200]  train-auc:0.751218+0.000592796  test-auc:0.731224+0.00280437
[1250]  train-auc:0.751949+0.000592142  test-auc:0.731339+0.00275725
[1300]  train-auc:0.752699+0.000574383  test-auc:0.731457+0.00272738
[1350]  train-auc:0.753465+0.000598359  test-auc:0.731582+0.00267593
[1400]  train-auc:0.754144+0.000597865  test-auc:0.731682+0.00267617
[1450]  train-auc:0.754824+0.000613228  test-auc:0.731756+0.00262873
[1500]  train-auc:0.755526+0.000606079  test-auc:0.731853+0.00262346
[1550]  train-auc:0.756213+0.000595768  test-auc:0.73194+0.00264273
[1600]  train-auc:0.756884+0.000612102  test-auc:0.732008+0.00263743
[1650]  train-auc:0.757514+0.000605446  test-auc:0.732111+0.00266223
[1700]  train-auc:0.758148+0.000596116  test-auc:0.732169+0.00266036
[1750]  train-auc:0.758769+0.000591895  test-auc:0.732239+0.00265111
[1800]  train-auc:0.759392+0.000570932  test-auc:0.732308+0.00264383
[1850]  train-auc:0.760006+0.00057841   test-auc:0.732375+0.00259932
[1900]  train-auc:0.760623+0.000616789  test-auc:0.732436+0.0025783
[1950]  train-auc:0.761194+0.000626881  test-auc:0.732463+0.00258138
[2000]  train-auc:0.761778+0.00063534   test-auc:0.7325+0.00258684
[2050]  train-auc:0.762362+0.00063628   test-auc:0.732544+0.00258746
[2100]  train-auc:0.762937+0.000636023  test-auc:0.732563+0.00256952
[2150]  train-auc:0.763457+0.00063863   test-auc:0.732592+0.0025644
[2200]  train-auc:0.764016+0.000641848  test-auc:0.732605+0.00256291
[2250]  train-auc:0.764582+0.000620005  test-auc:0.732666+0.00255352
[2300]  train-auc:0.765128+0.000591349  test-auc:0.732687+0.00255714
[2350]  train-auc:0.765709+0.00057196   test-auc:0.732714+0.00255496
[2400]  train-auc:0.766246+0.000570307  test-auc:0.732723+0.00252563
[2450]  train-auc:0.766748+0.000547467  test-auc:0.732735+0.00251021
[2500]  train-auc:0.767283+0.000552341  test-auc:0.732741+0.0025177
[2550]  train-auc:0.767835+0.000546115  test-auc:0.732727+0.00249082
[2600]  train-auc:0.768356+0.000559551  test-auc:0.732711+0.00249255
[2650]  train-auc:0.76886+0.00051772    test-auc:0.73272+0.00250602
Stopping. Best iteration:
[2488]  train-auc:0.767145+0.000546431  test-auc:0.732764+0.00250961

Best score 0.732764, best iteration 2489
```

Through cross-validation, the best iteration is found to be 2877, and the corresponding best AUC score is 0.7333. The XGBoost model is then trained on the training set, based on the best parameters and iteration.

In [164]:
```
xgb_final = xgb.train(best_params, dtrain, num_boost_round=best_iteration)
```

Model Integration

[go back to the top]

Pickle Model (xgb_bo) and Feature.

The trained XGBoost model object "xgb_final" here is saved into a pickle file named "model.pkl". A input file containing an example dataset is saved into a pickle file named "feature.pkl". Both model.pkl and feature.pkl are saved into a web-application framework build with Flask. Running the Flask framework results in a web application interface which allow users to adjust important parameters of a loan and get a predicted default rate.

In [165]:
```
import pickle
```

In [166]:
```
with open('model.pkl', 'wb') as outfile:
    pickle.dump(xgb_final, outfile)
```

Model Evaluation

[go back to the top]

Feature Importance

The importance of features is generated based on the feature importance scores from the XGBoost model. From the table and plot below it is found that the top 10 important features include Annual Income, Debt to Income Ratio, Interest Rate, Portion of Balance, and so on. Generally the important features match the trend observed in the Feature Responses Visualization.

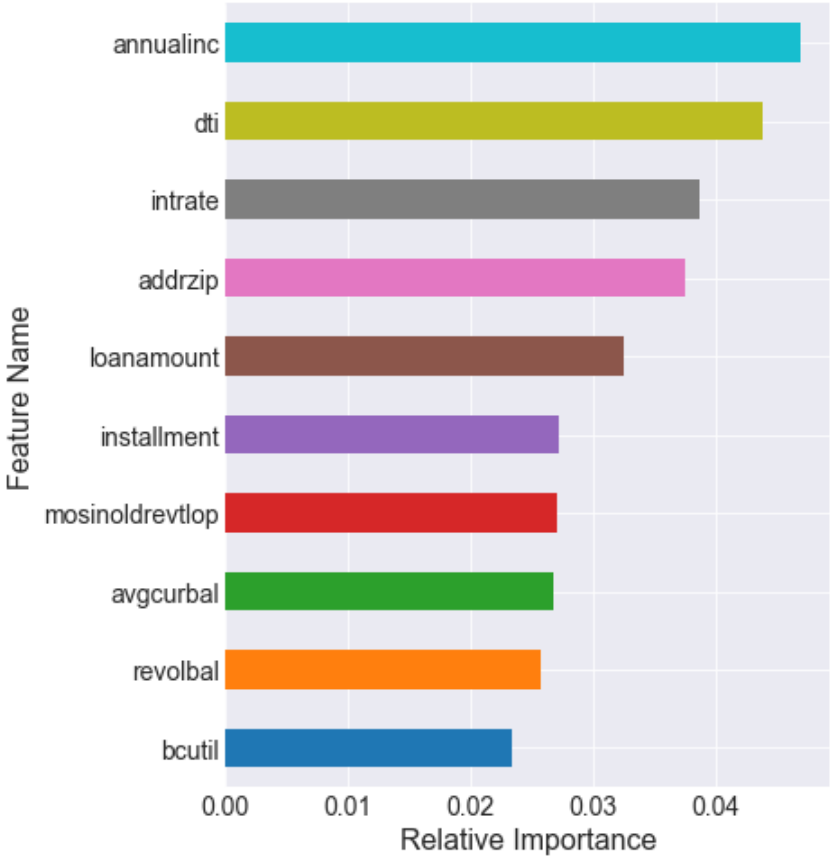
In [167]:
```
importance = xgb_final.get_fscore()
df_importance = pd.DataFrame(list(importance.items()), columns = ['feature', 'fscore'
])
df_importance['fscore'] = df_importance['fscore']/df_importance['fscore'].sum()
df_importance.sort_values(['fscore'], ascending= False, inplace= True)
```

In [168]:
```
df_importance.head(10)
```

Out[168]:

	feature	fscore
6	annualinc	0.046991
3	dti	0.043863
1	intrate	0.038671
20	addrzip	0.037473
7	loanamount	0.032548
12	installment	0.027290
15	mosinoldrevtlop	0.027023
4	avgcurbal	0.026824
25	revolbal	0.025759
37	bcutil	0.023363

In [169]:
```
df_importance[:10].sort_values(by='fscore').plot(kind = 'barh', x = 'feature', y = 'f
score', legend= False, figsize=(6,8) )
plt.xlabel('Relative Importance', fontsize = 16)
plt.ylabel('Feature Name', fontsize = 16)
plt.tick_params(labelsize = 14)
```



Testing Set Scores

The trained model is then used to predict the default rates in the testing dateset. The predicted default rate on the test set is about 0.2055, very close to the average default rate of around 0.20. The result of model performance is visualized via the ROC curve. The AUC scores on the training and validation sets are about 0.76, and the AUC score on the testing set is about 0.72.

In [170]:
```
y_pred = xgb_final.predict(dtest)
```

In [171]:
```
y_pred.mean(), y_pred.max(), y_pred.min()
```

Out[171]: (0.21466491, 0.88088685, 0.0057469923)

In [172]:
```
df_2014.loanstatus.mean()
```

Out[172]: 0.20513350051630033

Train/Test Split in x_train

In [173]:
```
from sklearn.cross_validation import StratifiedKFold
from sklearn.cross_validation import train_test_split
```

In [174]:
```
x_t, x_cv, y_t, y_cv = train_test_split(x_train, y_train, test_size = 0.3, \
                                        random_state = 1234, stratify = y_train)
```

Plot ROC-AUC

In [175]:
```
from sklearn.metrics import roc_curve, auc
from sklearn import linear_model, datasets
```

In [176]:
```
type(y_t)
```

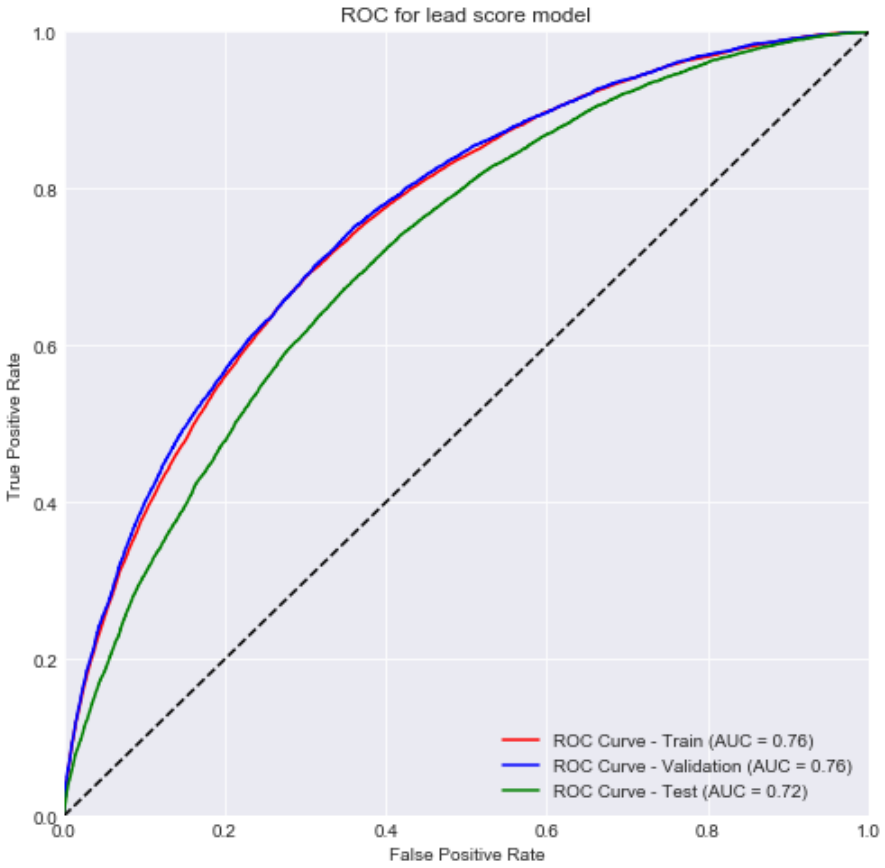
Out[176]:
```
pandas.core.series.Series
```

In [177]:
```
from sklearn.metrics import roc_curve, auc
from sklearn import linear_model, datasets

def draw_ROC(model, traindata, cvdata, testdata, trainy, cvy, testy):
    prob_0 = model.predict(traindata)
    prob_1 = model.predict(cvdata)
    prob_2 = model.predict(testdata)
    fpr_0, tpr_0, thresholds_0 = roc_curve(trainy, prob_0)
    fpr_1, tpr_1, thresholds_1 = roc_curve(cvy, prob_1)
    fpr_2, tpr_2, thresholds_2 = roc_curve(testy, prob_2)
    roc_auc_0 = auc(fpr_0, tpr_0)
    roc_auc_1 = auc(fpr_1, tpr_1)
    roc_auc_2 = auc(fpr_2, tpr_2)

    print ("Area under the ROC Curve - Train: %f" %(roc_auc_0))
    print ("Area under the ROC Curve - Validation: %f" % (roc_auc_1))
    print ("Area under the ROC Curve - Test: %f" % (roc_auc_2))


    #Plot ROC Curve
    plt.figure(figsize = (8,8))
    plt.plot(fpr_0, tpr_0, label = 'ROC Curve - Train (AUC = %0.2f)' % (roc_auc_0), c
olor= 'r')
    plt.plot(fpr_1, tpr_1, label = 'ROC Curve - Validation (AUC = %0.2f)' % (roc_auc_
1), color= 'b')
    plt.plot(fpr_2, tpr_2, label = 'ROC Curve - Test (AUC = %0.2f)' % (roc_auc_2), co
lor= 'g')
    plt.plot([0,1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.0])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC for lead score model')
    plt.legend(loc='lower right')
    plt.show()
```

In [178]:
```
x_t_xgb = xgb.DMatrix(x_t, missing= np.nan)
x_cv_xgb = xgb.DMatrix(x_cv, missing= np.nan)
```

In [179]:
```
draw_ROC(xgb_final, x_t_xgb, x_cv_xgb, dtest, y_t, y_cv, y_test)
```

```
Area under the ROC Curve - Train: 0.760634
Area under the ROC Curve - Validation: 0.764565
Area under the ROC Curve - Test: 0.720915
```



CONCLUSION

[go back to the top]

This FinTech project built a program that serves as an intelligent investment consultant, which in estabishes a predictive model to evaluate the outcome of loan default rate in real-time, and feedback the prediction results to users via an interactive web-application portal. The model is built upon the historic loan data of year 2014 from the Lending Club's open resource. After data cleaning and feature engineering, model is built based on XGBoost algorithm with AUC score as model metrics. Bayesian Optimization is used to maximize the AUC score and find out the best parameters for the XGBoost model. Finally, a well-tuned predictive model is established with a AUC score of 0.73 on test sets. Feature importance plot shows several key features in predicting the default rate, such as Debt-to-Income Ratio, Annual Income, Portion of Balance, Interest Rate, and so on.