Part I - Loan Data from Prosper dataset exploration

by Muyul Alsubaie

> This data set contains 113,937 loans with 81 variables on each loan, including loan amount, borrower rate (or interest rate), current loan status, borrower income, and many others.

Preliminary Wrangling

In [4]:
```
# import all packages and set plots to be embedded inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
import seaborn as sb

%matplotlib inline
```

> Load in your dataset and describe its properties through the questions below. Try and motivate your exploration goals through this section.

In [5]:
```
# load the dataset
PL = pd.read_csv('prosperLoanData.csv')
```

In [6]:
```
pd.set_option('display.max_columns', None) #to display all columns
PL.head()
```

Out[6]:

	ListingKey	ListingNumber	ListingCreationDate	CreditGrade	Term	LoanStatus	ClosedDate	Bo
0	1021339766868145413AB3B	193129	2007-08-26 19:09:29.263000000	C	36	Completed	2009-08-14 00:00:00	
1	10273602499503308B223C1	1209647	2014-02-27 08:28:07.900000000	NaN	36	Current	NaN	
2	0EE9337825851032864889A	81716	2007-01-05 15:00:47.090000000	HR	36	Completed	2009-12-17 00:00:00	
3	0EF5356002482715299901A	658116	2012-10-22 11:02:35.010000000	NaN	36	Current	NaN	
4	0F023589499656230C5E3E2	909464	2013-09-14 18:38:39.097000000	NaN	36	Current	NaN	

In [7]:
```
# descriptive statistics for numeric variables
print(PL.describe())
```

```
       ListingNumber           Term     BorrowerAPR    BorrowerRate  \
count   1.139370e+05  113937.000000  113912.000000  113937.000000
mean    6.278857e+05      40.830248       0.218828       0.192764
std     3.280762e+05      10.436212       0.080364       0.074818
min     4.000000e+00      12.000000       0.006530       0.000000
25%     4.009190e+05      36.000000       0.156290       0.134000
50%     6.005540e+05      36.000000       0.209760       0.184000
```

75%	8.926340e+05	36.000000	0.283810	0.250000
max	1.255725e+06	60.000000	0.512290	0.497500

	LenderYield	EstimatedEffectiveYield	EstimatedLoss	EstimatedReturn \
count	113937.000000	84853.000000	84853.000000	84853.000000
mean	0.182701	0.168661	0.080306	0.096068
std	0.074516	0.068467	0.046764	0.030403
min	-0.010000	-0.182700	0.004900	-0.182700
25%	0.124200	0.115670	0.042400	0.074080
50%	0.173000	0.161500	0.072400	0.091700
75%	0.240000	0.224300	0.112000	0.116600
max	0.492500	0.319900	0.366000	0.283700

	ProsperRating (numeric)	ProsperScore	ListingCategory (numeric) \
count	84853.000000	84853.000000	113937.000000
mean	4.072243	5.950067	2.774209
std	1.673227	2.376501	3.996797
min	1.000000	1.000000	0.000000
25%	3.000000	4.000000	1.000000
50%	4.000000	6.000000	1.000000
75%	5.000000	8.000000	3.000000
max	7.000000	11.000000	20.000000

	EmploymentStatusDuration	CreditScoreRangeLower	CreditScoreRangeUpper \
count	106312.000000	113346.000000	113346.000000
mean	96.071582	685.567731	704.567731
std	94.480605	66.458275	66.458275
min	0.000000	0.000000	19.000000
25%	26.000000	660.000000	679.000000
50%	67.000000	680.000000	699.000000
75%	137.000000	720.000000	739.000000
max	755.000000	880.000000	899.000000

	CurrentCreditLines	OpenCreditLines	TotalCreditLinespast7years \
count	106333.000000	106333.000000	113240.000000
mean	10.317192	9.260164	26.754539
std	5.457866	5.022644	13.637871
min	0.000000	0.000000	2.000000
25%	7.000000	6.000000	17.000000
50%	10.000000	9.000000	25.000000
75%	13.000000	12.000000	35.000000
max	59.000000	54.000000	136.000000

	OpenRevolvingAccounts	OpenRevolvingMonthlyPayment \
count	113937.00000	113937.000000
mean	6.96979	398.292161
std	4.63097	447.159711
min	0.00000	0.000000
25%	4.00000	114.000000
50%	6.00000	271.000000
75%	9.00000	525.000000
max	51.00000	14985.000000

	InquiriesLast6Months	TotalInquiries	CurrentDelinquencies \
count	113240.000000	112778.000000	113240.000000
mean	1.435085	5.584405	0.592052
std	2.437507	6.429946	1.978707
min	0.000000	0.000000	0.000000
25%	0.000000	2.000000	0.000000
50%	1.000000	4.000000	0.000000
75%	2.000000	7.000000	0.000000
max	105.000000	379.000000	83.000000

	AmountDelinquent	DelinquenciesLast7Years	PublicRecordsLast10Years \
count	106315.000000	112947.000000	113240.000000
mean	984.507059	4.154984	0.312646

```
std      7158.270157                    10.160216                          0.727868
min         0.000000                     0.000000                          0.000000
25%         0.000000                     0.000000                          0.000000
50%         0.000000                     0.000000                          0.000000
75%         0.000000                     3.000000                          0.000000
max    463881.000000                    99.000000                         38.000000

       PublicRecordsLast12Months  RevolvingCreditBalance  BankcardUtilization  \
count              106333.000000            1.063330e+05        106333.000000   
mean                    0.015094            1.759871e+04             0.561309   
std                     0.154092            3.293640e+04             0.317918   
min                     0.000000            0.000000e+00             0.000000   
25%                     0.000000            3.121000e+03             0.310000   
50%                     0.000000            8.549000e+03             0.600000   
75%                     0.000000            1.952100e+04             0.840000   
max                    20.000000            1.435667e+06             5.950000   

       AvailableBankcardCredit    TotalTrades  \
count            106393.000000  106393.000000   
mean              11210.225447      23.230034   
std               19818.361309      11.871311   
min                   0.000000       0.000000   
25%                 880.000000      15.000000   
50%                4100.000000      22.000000   
75%               13180.000000      30.000000   
max              646285.000000     126.000000   

       TradesNeverDelinquent (percentage)  TradesOpenedLast6Months  \
count                       106393.000000            106393.000000   
mean                             0.885897                 0.802327   
std                              0.148179                 1.097637   
min                              0.000000                 0.000000   
25%                              0.820000                 0.000000   
50%                              0.940000                 0.000000   
75%                              1.000000                 1.000000   
max                              1.000000                20.000000   

       DebtToIncomeRatio  StatedMonthlyIncome  TotalProsperLoans  \
count      105383.000000         1.139370e+05       22085.000000   
mean            0.275947         5.608026e+03           1.421100   
std             0.551759         7.478497e+03           0.764042   
min             0.000000         0.000000e+00           0.000000   
25%             0.140000         3.200333e+03           1.000000   
50%             0.220000         4.666667e+03           1.000000   
75%             0.320000         6.825000e+03           2.000000   
max            10.010000         1.750003e+06           8.000000   

       TotalProsperPaymentsBilled  OnTimeProsperPayments  \
count                22085.000000           22085.000000   
mean                    22.934345              22.271949   
std                     19.249584              18.830425   
min                      0.000000               0.000000   
25%                      9.000000               9.000000   
50%                     16.000000              15.000000   
75%                     33.000000              32.000000   
max                    141.000000             141.000000   

       ProsperPaymentsLessThanOneMonthLate  ProsperPaymentsOneMonthPlusLate  \
count                         22085.000000                     22085.000000   
mean                              0.613629                         0.048540   
std                               2.446827                         0.556285   
min                               0.000000                         0.000000   
25%                               0.000000                         0.000000   
50%                               0.000000                         0.000000   
75%                               0.000000                         0.000000   
max                              42.000000                        21.000000   
```

```
       ProsperPrincipalBorrowed  ProsperPrincipalOutstanding  \
count              22085.000000                 22085.000000   
mean                8472.311961                  2930.313906   
std                 7395.507650                  3806.635075   
min                    0.000000                     0.000000   
25%                 3500.000000                     0.000000   
50%                 6000.000000                  1626.550000   
75%                11000.000000                  4126.720000   
max                72499.000000                 23450.950000   

       ScorexChangeAtTimeOfListing  LoanCurrentDaysDelinquent  \
count                 18928.000000              113937.000000   
mean                     -3.223214                 152.816539   
std                      50.063567                 466.320254   
min                    -209.000000                   0.000000   
25%                     -35.000000                   0.000000   
50%                      -3.000000                   0.000000   
75%                      25.000000                   0.000000   
max                     286.000000                2704.000000   

       LoanFirstDefaultedCycleNumber  LoanMonthsSinceOrigination  \
count                   16952.000000               113937.000000   
mean                       16.268464                   31.896882   
std                         9.005898                   29.974184   
min                         0.000000                    0.000000   
25%                         9.000000                    6.000000   
50%                        14.000000                   21.000000   
75%                        22.000000                   65.000000   
max                        44.000000                  100.000000   

          LoanNumber  LoanOriginalAmount  MonthlyLoanPayment  \
count  113937.000000        113937.00000       113937.000000   
mean    69444.474271          8337.01385          272.475783   
std     38930.479610          6245.80058          192.697812   
min         1.000000          1000.00000            0.000000   
25%     37332.000000          4000.00000          131.620000   
50%     68599.000000          6500.00000          217.740000   
75%    101901.000000         12000.00000          371.580000   
max    136486.000000         35000.00000         2251.510000   

       LP_CustomerPayments  LP_CustomerPrincipalPayments  LP_InterestandFees  \
count        113937.000000                 113937.000000       113937.000000   
mean           4183.079489                   3105.536588         1077.542901   
std            4790.907234                   4069.527670         1183.414168   
min              -2.349900                      0.000000           -2.349900   
25%            1005.760000                    500.890000          274.870000   
50%            2583.830000                   1587.500000          700.840100   
75%            5548.400000                   4000.000000         1458.540000   
max           40702.390000                  35000.000000        15617.030000   

       LP_ServiceFees  LP_CollectionFees  LP_GrossPrincipalLoss  \
count   113937.000000      113937.000000          113937.000000   
mean       -54.725641         -14.242698             700.446342   
std         60.675425         109.232758            2388.513831   
min       -664.870000       -9274.750000             -94.200000   
25%        -73.180000           0.000000               0.000000   
50%        -34.440000           0.000000               0.000000   
75%        -13.920000           0.000000               0.000000   
max         32.060000           0.000000           25000.000000   

       LP_NetPrincipalLoss  LP_NonPrincipalRecoverypayments  PercentFunded  \
count        113937.000000                    113937.000000  113937.000000   
mean            681.420499                        25.142686       0.998584   
std            2357.167068                       275.657937       0.017919   
min            -954.550000                         0.000000       0.700000   
```

```
25%              0.000000                              0.000000        1.000000
50%              0.000000                              0.000000        1.000000
75%              0.000000                              0.000000        1.000000
max          25000.000000                          21117.900000        1.012500

       Recommendations  InvestmentFromFriendsCount  \
count    113937.000000               113937.000000
mean          0.048027                    0.023460
std           0.332353                    0.232412
min           0.000000                    0.000000
25%           0.000000                    0.000000
50%           0.000000                    0.000000
75%           0.000000                    0.000000
max          39.000000                   33.000000

       InvestmentFromFriendsAmount      Investors
count                113937.000000  113937.000000
mean                     16.550751      80.475228
std                     294.545422     103.239020
min                       0.000000       1.000000
25%                       0.000000       2.000000
50%                       0.000000      44.000000
75%                       0.000000     115.000000
max                   25000.000000    1189.000000
```

In [8]:
```
PL.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 113937 entries, 0 to 113936
Data columns (total 81 columns):
 #   Column                               Non-Null Count   Dtype  
---  ------                               --------------   -----  
 0   ListingKey                           113937 non-null  object 
 1   ListingNumber                        113937 non-null  int64  
 2   ListingCreationDate                  113937 non-null  object 
 3   CreditGrade                          28953 non-null   object 
 4   Term                                 113937 non-null  int64  
 5   LoanStatus                           113937 non-null  object 
 6   ClosedDate                           55089 non-null   object 
 7   BorrowerAPR                          113912 non-null  float64
 8   BorrowerRate                         113937 non-null  float64
 9   LenderYield                          113937 non-null  float64
 10  EstimatedEffectiveYield              84853 non-null   float64
 11  EstimatedLoss                        84853 non-null   float64
 12  EstimatedReturn                      84853 non-null   float64
 13  ProsperRating (numeric)              84853 non-null   float64
 14  ProsperRating (Alpha)                84853 non-null   object 
 15  ProsperScore                         84853 non-null   float64
 16  ListingCategory (numeric)            113937 non-null  int64  
 17  BorrowerState                        108422 non-null  object 
 18  Occupation                           110349 non-null  object 
 19  EmploymentStatus                     111682 non-null  object 
 20  EmploymentStatusDuration             106312 non-null  float64
 21  IsBorrowerHomeowner                  113937 non-null  bool   
 22  CurrentlyInGroup                     113937 non-null  bool   
 23  GroupKey                             13341 non-null   object 
 24  DateCreditPulled                     113937 non-null  object 
 25  CreditScoreRangeLower                113346 non-null  float64
 26  CreditScoreRangeUpper                113346 non-null  float64
 27  FirstRecordedCreditLine              113240 non-null  object 
 28  CurrentCreditLines                   106333 non-null  float64
 29  OpenCreditLines                      106333 non-null  float64
 30  TotalCreditLinespast7years           113240 non-null  float64
 31  OpenRevolvingAccounts                113937 non-null  int64  
 32  OpenRevolvingMonthlyPayment          113937 non-null  float64
 33  InquiriesLast6Months                 113240 non-null  float64
```

```
 34  TotalInquiries                       112778 non-null  float64
 35  CurrentDelinquencies                 113240 non-null  float64
 36  AmountDelinquent                     106315 non-null  float64
 37  DelinquenciesLast7Years              112947 non-null  float64
 38  PublicRecordsLast10Years             113240 non-null  float64
 39  PublicRecordsLast12Months            106333 non-null  float64
 40  RevolvingCreditBalance               106333 non-null  float64
 41  BankcardUtilization                  106333 non-null  float64
 42  AvailableBankcardCredit              106393 non-null  float64
 43  TotalTrades                          106393 non-null  float64
 44  TradesNeverDelinquent (percentage)   106393 non-null  float64
 45  TradesOpenedLast6Months              106393 non-null  float64
 46  DebtToIncomeRatio                    105383 non-null  float64
 47  IncomeRange                          113937 non-null  object
 48  IncomeVerifiable                     113937 non-null  bool
 49  StatedMonthlyIncome                  113937 non-null  float64
 50  LoanKey                              113937 non-null  object
 51  TotalProsperLoans                    22085 non-null   float64
 52  TotalProsperPaymentsBilled           22085 non-null   float64
 53  OnTimeProsperPayments                22085 non-null   float64
 54  ProsperPaymentsLessThanOneMonthLate  22085 non-null   float64
 55  ProsperPaymentsOneMonthPlusLate      22085 non-null   float64
 56  ProsperPrincipalBorrowed             22085 non-null   float64
 57  ProsperPrincipalOutstanding          22085 non-null   float64
 58  ScorexChangeAtTimeOfListing          18928 non-null   float64
 59  LoanCurrentDaysDelinquent            113937 non-null  int64
 60  LoanFirstDefaultedCycleNumber        16952 non-null   float64
 61  LoanMonthsSinceOrigination           113937 non-null  int64
 62  LoanNumber                           113937 non-null  int64
 63  LoanOriginalAmount                   113937 non-null  int64
 64  LoanOriginationDate                  113937 non-null  object
 65  LoanOriginationQuarter               113937 non-null  object
 66  MemberKey                            113937 non-null  object
 67  MonthlyLoanPayment                   113937 non-null  float64
 68  LP_CustomerPayments                  113937 non-null  float64
 69  LP_CustomerPrincipalPayments         113937 non-null  float64
 70  LP_InterestandFees                   113937 non-null  float64
 71  LP_ServiceFees                       113937 non-null  float64
 72  LP_CollectionFees                    113937 non-null  float64
 73  LP_GrossPrincipalLoss                113937 non-null  float64
 74  LP_NetPrincipalLoss                  113937 non-null  float64
 75  LP_NonPrincipalRecoverypayments      113937 non-null  float64
 76  PercentFunded                        113937 non-null  float64
 77  Recommendations                      113937 non-null  int64
 78  InvestmentFromFriendsCount           113937 non-null  int64
 79  InvestmentFromFriendsAmount          113937 non-null  float64
 80  Investors                            113937 non-null  int64
dtypes: bool(3), float64(50), int64(11), object(17)
memory usage: 68.1+ MB
```

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

Out[9]:
```
(113937, 81)
```

In [10]:
```
PL.dtypes
```

Out[10]:
```
ListingKey                     object
ListingNumber                   int64
ListingCreationDate            object
CreditGrade                    object
Term                            int64
                                ...
PercentFunded                 float64
Recommendations                 int64
InvestmentFromFriendsCount      int64
```

```
InvestmentFromFriendsAmount    float64
Investors                        int64
Length: 81, dtype: object
```

What is the structure of your dataset?

> The dataset contains 113,937 loans with 81 features, most varviables are numeric.

What is/are the main feature(s) of interest in your dataset?

> I'm intrested in the borrower's Annual Percentage Rate (APR) for the loan, and Which lender features are most predictive of the highest rate of return.

What features in the dataset do you think will help support your investigation into your feature(s) of interest?

> EstimatedReturn, IncomeRange,Debt to Income Ratio, and BorrowerAPR.

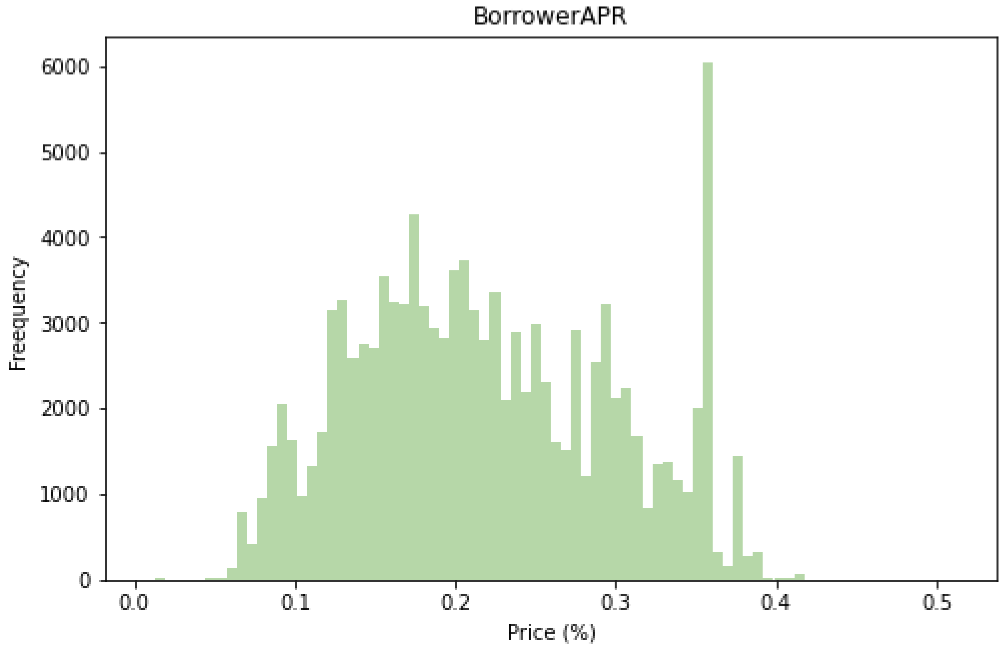
Univariate Exploration

BorrowerAPR

> I'll start by looking at the distribution of the main variable of interest: borrower APR.

In [11]:
```
#A custom method to display all the required plots
def display_plot(variable, xlabel,title):

    # Method will take four parameters the first two variables are for the used data
    #2nd, 3rd and 4th parameters are to determine the axes names and title
    plot.figure(figsize=[8,5])
    variable.plot(kind='hist',color='#b6d7a8',bins=80)
    plot.xlabel(' ')
    plot.title('');
```

In [12]:
```
#bins = np.arange(0, PL['BorrowerAPR'].max()+binsize, binsize)
display_plot( PL['BorrowerAPR'], 'Price (%)', 'BorrowerAPR')
plot.xlabel('Price (%)')
plot.ylabel('Freequency ')
plot.title('BorrowerAPR');
```



> There is a narrow rise at 0.9 and a small low point centered 0.28, there is a significant high point at 0.2, as well as a high point between 0.34 and 0.36, and only a few loans have an APR greater than 0.42%.

In [13]:
```
# loans with APR greater than 0.42
PL[PL.BorrowerAPR>0.42]
```

Out[13]:

	ListingKey	ListingNumber	ListingCreationDate	CreditGrade	Term	LoanStatus	ClosedDa
18326	0161336483146123835D6A5	1795	2006-03-11 15:43:45.393000000	HR	36	Defaulted	2007-01- 00:00:
22195	5686336572505607862C0C7	1849	2006-03-12 13:44:15.060000000	HR	36	Chargedoff	2009-02- 00:00:
36018	844033650124564886B3EDC	690	2006-02-23 13:57:02.087000000	HR	36	Completed	2006-03- 00:00:
56761	A79D33661366830833F3EF5	2231	2006-03-16 19:30:16.753000000	HR	36	Defaulted	2006-09- 00:00:
82043	BBED336465905564254DC8B	1112	2006-03-02 19:00:17.593000000	HR	36	Defaulted	2006-09- 00:00:
103973	95ED3365915044756AB754F	1366	2006-03-06 22:36:53.753000000	HR	36	Defaulted	2006-10- 00:00:
105889	CC0C3497369291932E3CF0E	481141	2010-10-22 14:07:40.683000000	NaN	36	Chargedoff	2011-04- 00:00:

> There are no Prosper rating or employment status records for the six borrowers with the highest APR.

Estimated Return

In [14]:
```
PL.EstimatedReturn.isnull().sum() #checking for the sum of null values
```

Out[14]: 29084

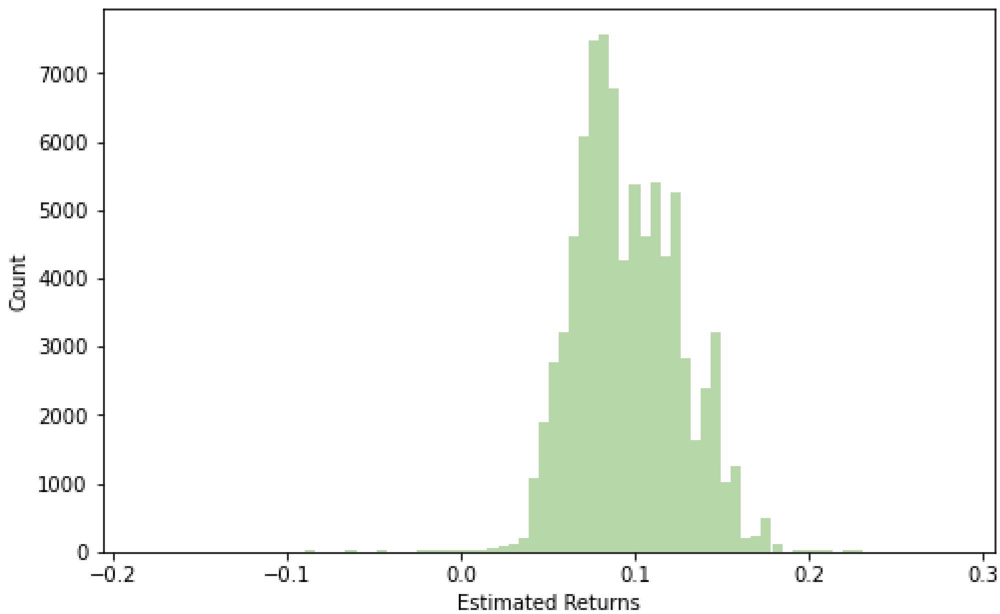
In [15]:
```
#drop null rows
PL.EstimatedReturn.dropna(axis = 0, inplace = True)
```

In [16]:
```
PL.EstimatedReturn.describe()
```

Out[16]:
```
count    84853.000000
mean         0.096068
std          0.030403
min         -0.182700
25%          0.074080
50%          0.091700
75%          0.116600
max          0.283700
Name: EstimatedReturn, dtype: float64
```

> The range of estimated returns is -18% to 28%.

In [17]:
```
# plot histogram
display_plot( PL['EstimatedReturn'], 'Estimated Returns',  'BorrowerAPR')
plot.xlabel('Estimated Returns')
plot.ylabel('Count');
```

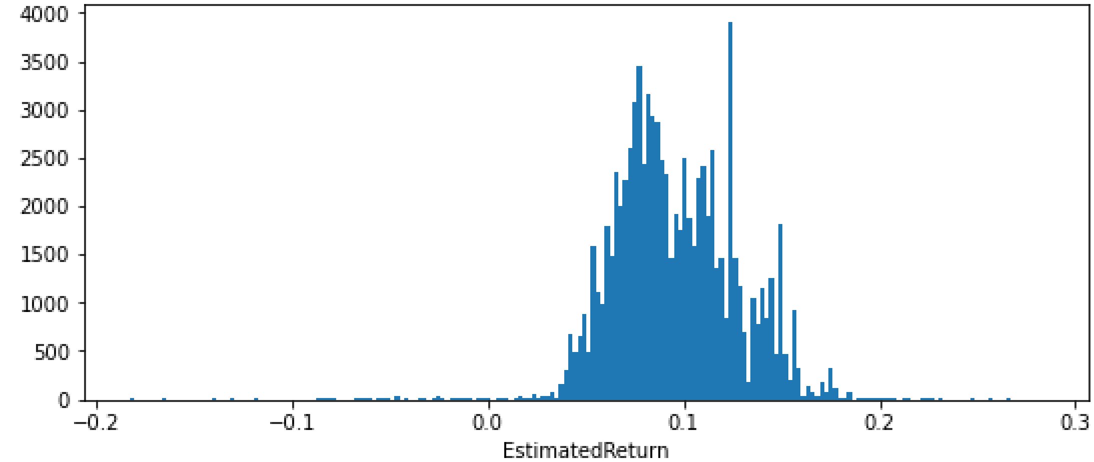


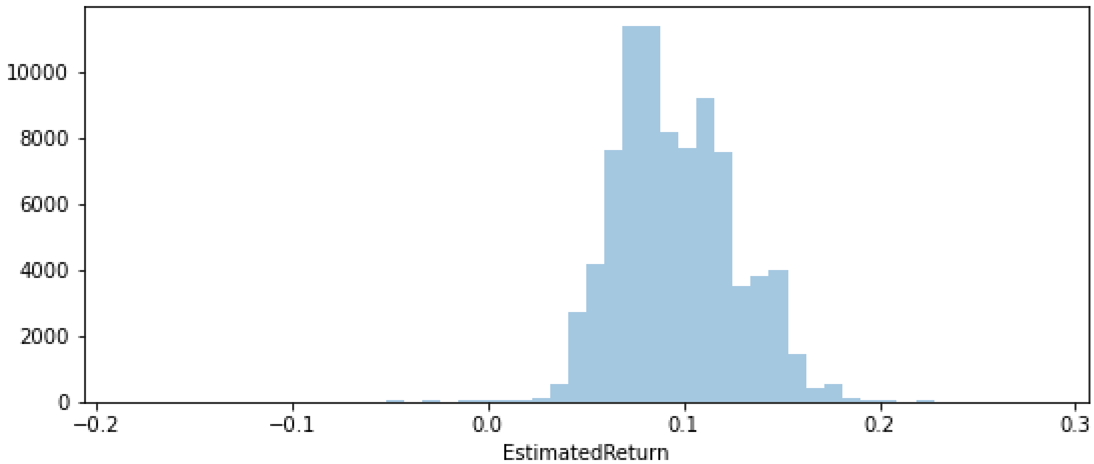
> Loans have an estimated return between 0% and 20%.

In [18]:
```
# ploting EstimatedReturn
fig, ax = plot.subplots(nrows=2, figsize = [9,8])
variable = ['EstimatedReturn']
for i in range(len(variable)):
    var = variable[i]
    sb.distplot(PL.EstimatedReturn,kde = False  )
    ax[i].hist(data = PL, x = var, bins = 200)
    ax[i].set_xlabel('{}'.format(var))

plot.show()
```

```
D:\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot`
is a deprecated function and will be removed in a future version. Please adapt your code
to use either `displot` (a figure-level function with similar flexibility) or `histplot`
```

```
(an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





> Surprisingly, the largest bin is around 12.5% and there are a few maxima in some standard values, such as 5.3%, 7.4%, and 15%.

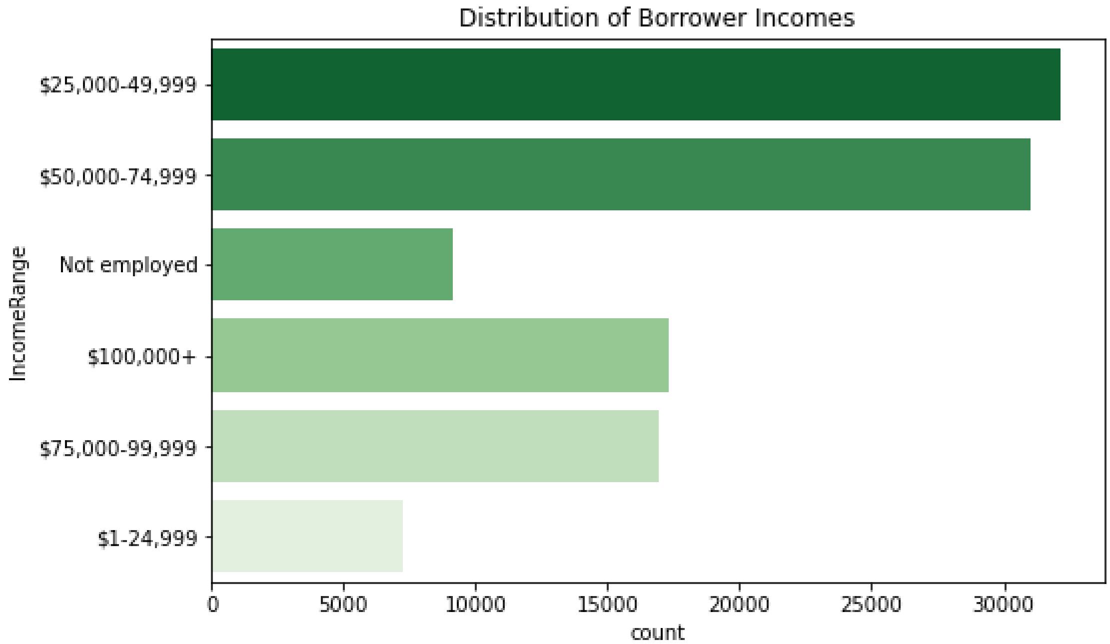
IncomeRange

In [19]:
```
PL.IncomeRange.value_counts()
```

Out[19]:
```
$25,000-49,999    32192
$50,000-74,999    31050
$100,000+         17337
$75,000-99,999    16916
Not displayed      7741
$1-24,999          7274
Not employed        806
$0                  621
Name: IncomeRange, dtype: int64
```

In [20]:
```
# Replacing not displayed in 0$ income with Not employed
PL['IncomeRange'].replace(['$0', 'Not displayed'], 'Not employed', inplace = True)
```

In [21]:
```
plot.figure(figsize = [8, 5])
sb.countplot(data=PL, y='IncomeRange',palette ='Greens_r')
plot.title('Distribution of Borrower Incomes');
```



> The majority of loan requests come from borrowers with incomes of between 25,000k and 49,000k.

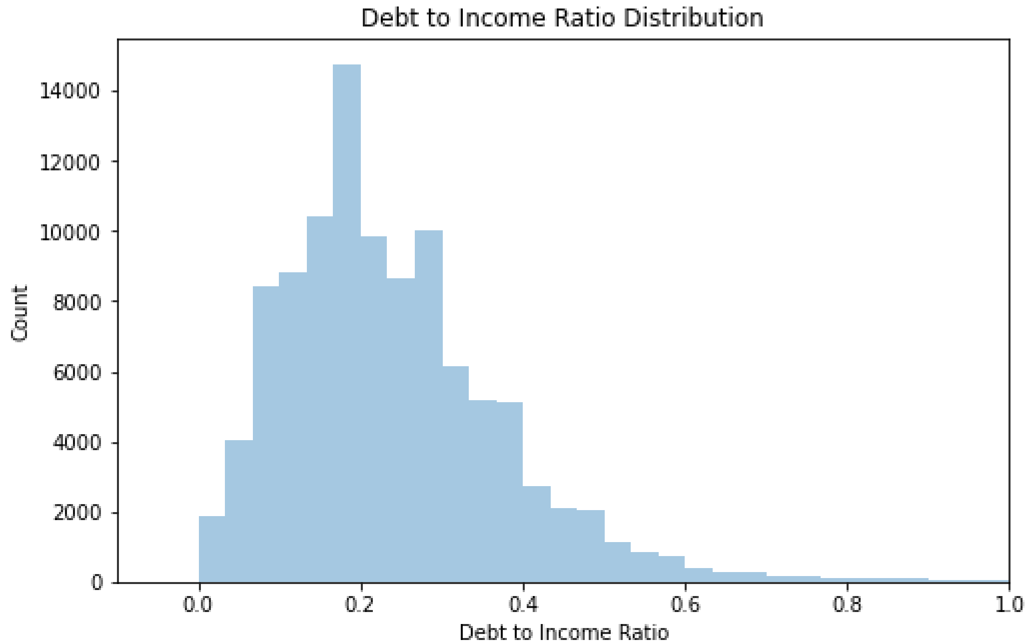
Debt to Income Ratio

In [22]:
```
PL.DebtToIncomeRatio.describe()
```

Out[22]:
```
count    105383.000000
mean          0.275947
std           0.551759
min           0.000000
25%           0.140000
50%           0.220000
75%           0.320000
max          10.010000
Name: DebtToIncomeRatio, dtype: float64
```

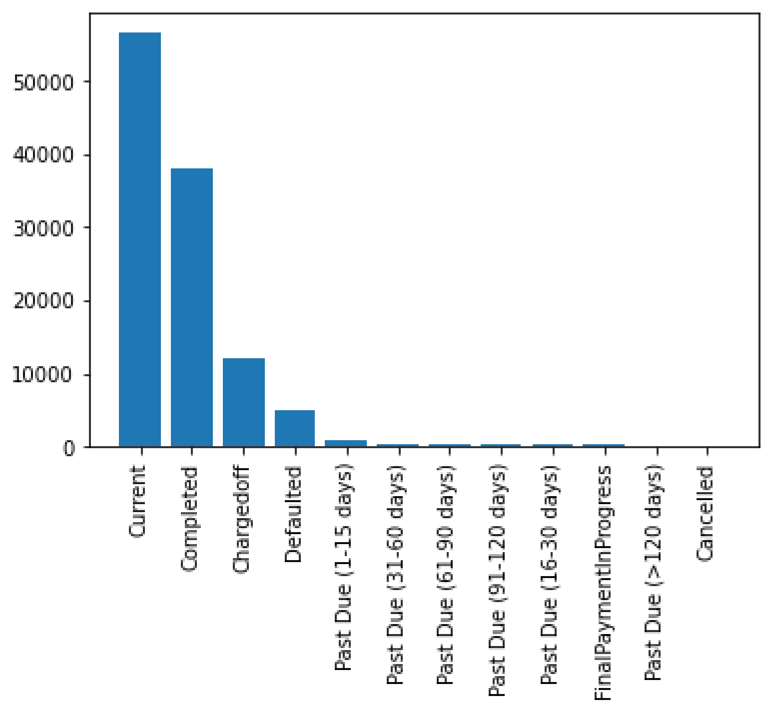
In [23]:
```
# distribution plot
plot.figure(figsize = [8,5])
sb.distplot(PL.DebtToIncomeRatio, kde = False, bins = 300)
plot.xlim(-.1, 1)
plot.xlabel('Debt to Income Ratio')
plot.ylabel('Count')
plot.title('Debt to Income Ratio Distribution');
```

```
D:\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



> as shown, borrowers who request a loan have a 20% debt to income ratio.

In [24]:
```
# Checking for the loan status
LoanStat = PL['LoanStatus'].value_counts()
plot.bar(LoanStat.index, LoanStat)
plot.xticks(rotation = 90);
```



The highest rate is for borrowers who are still paying their loan, followed by borrowers who completed paying for their loan.

Discuss the distribution(s) of your variable(s) of interest. Were there any unusual points? Did you need to perform any transformations?

> There are no unusual points and no need to perform any transformations.

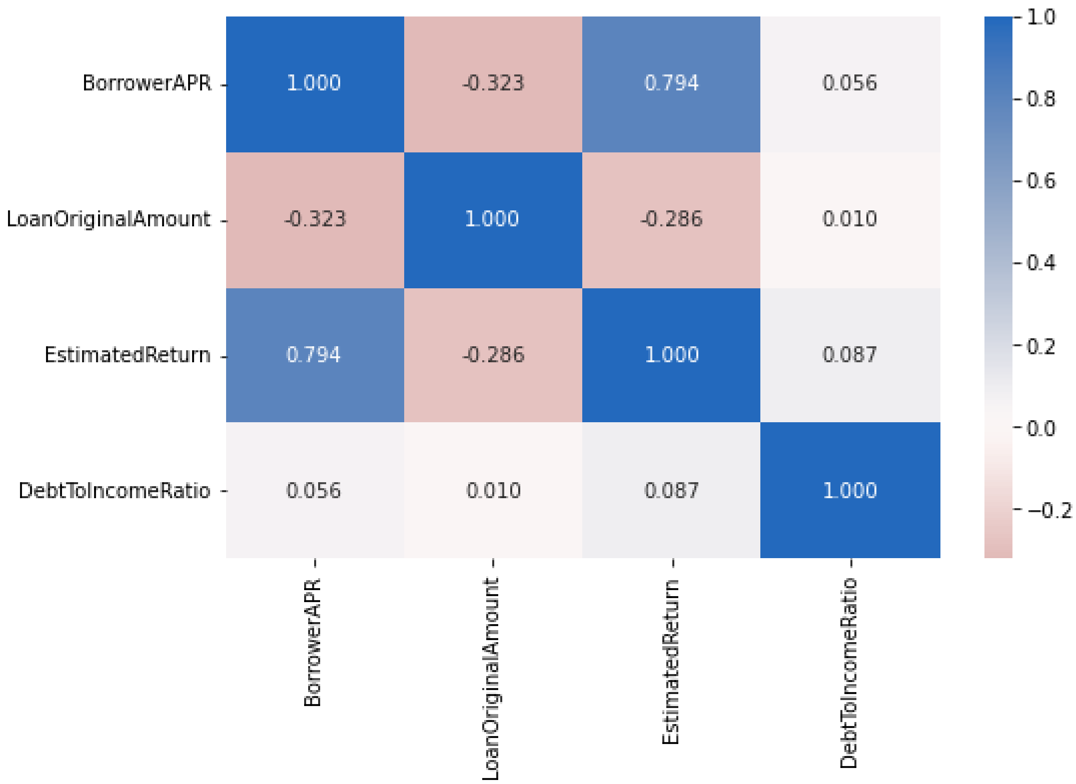
Of the features you investigated, were there any unusual distributions? Did you perform any operations on the data to tidy, adjust, or change the form of the data? If so, why did you do this?

> There was no unusual disributions.

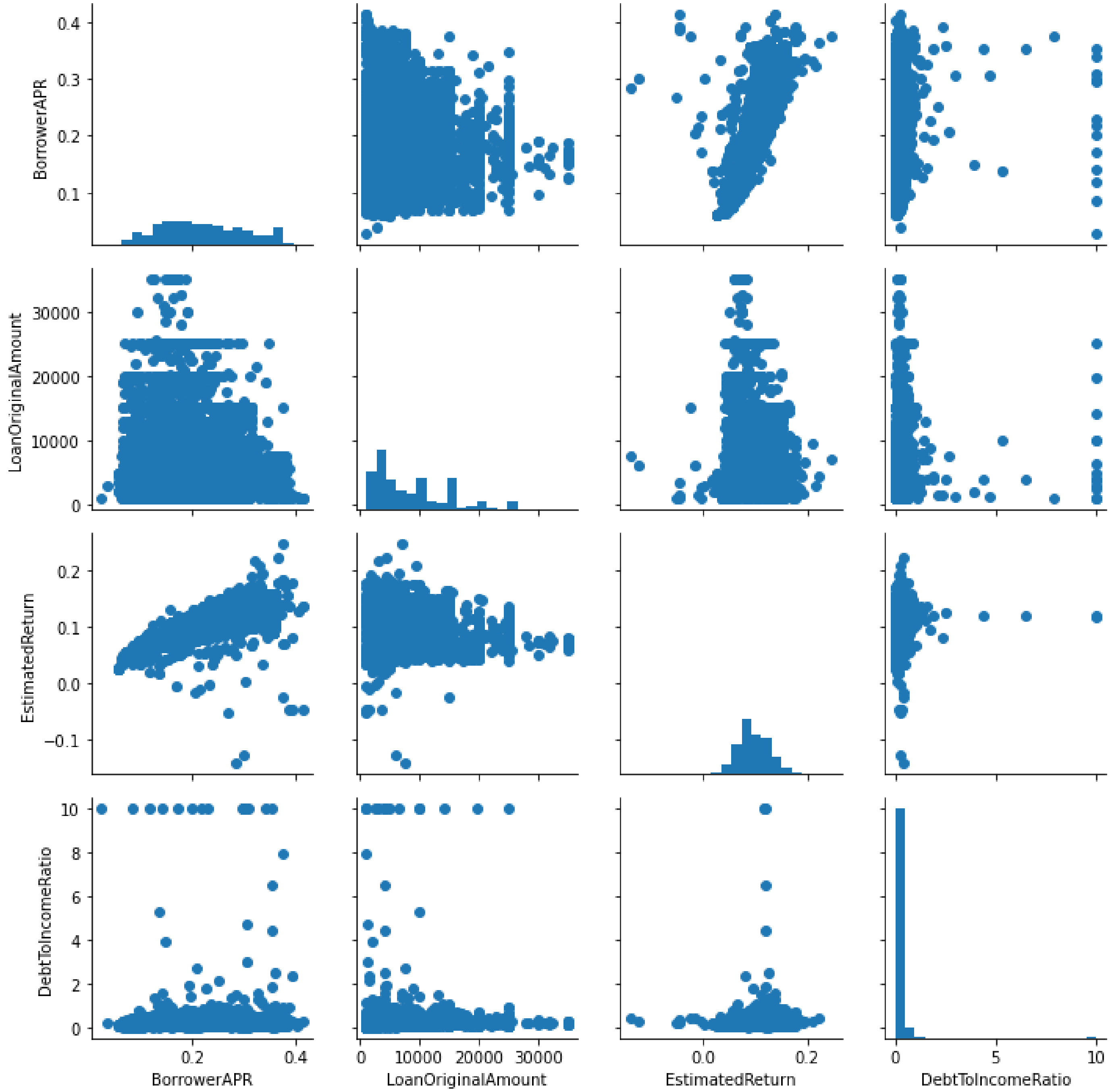
Bivariate Exploration

In [25]:
```
num_variabls = [ 'BorrowerAPR', 'LoanOriginalAmount','EstimatedReturn','DebtToIncomeRati
cat_variabls  = ['EmploymentStatus','Term', 'ProsperRating (Alpha)']
```

In [26]:
```
# correlation plot
plot.figure(figsize = [8, 5])
sb.heatmap(PL[num_variabls].corr(), annot = True, fmt = '.3f',
           cmap = 'vlag_r', center = 0);
```



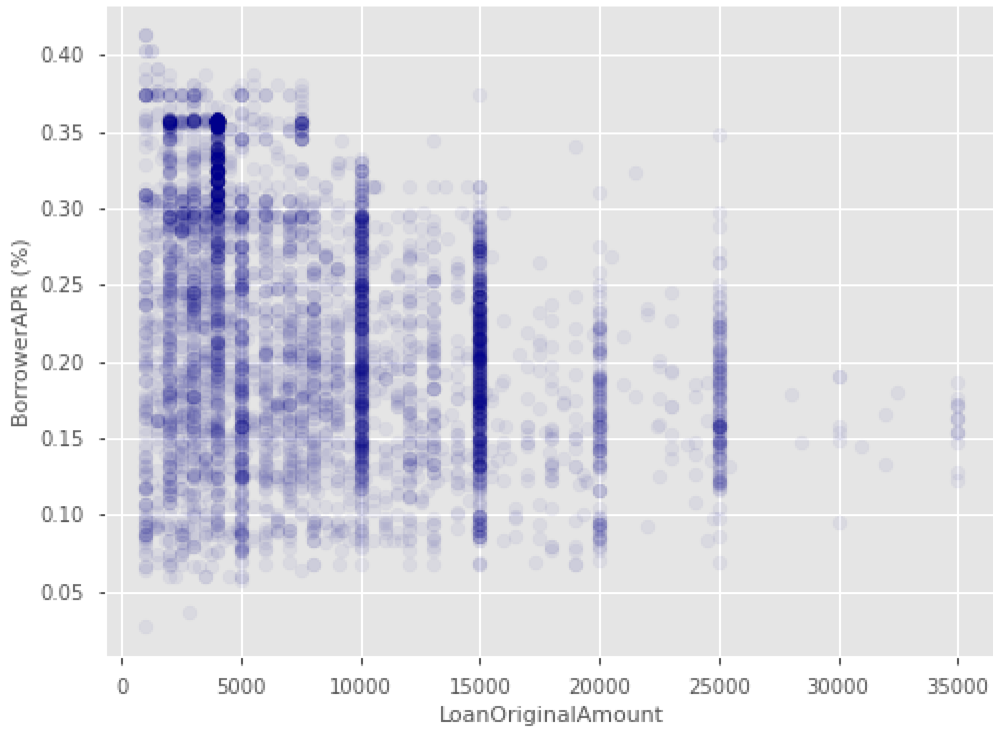
In [27]:
```
# plot matrix: sample 5000 loans so that plots are clearer

PL_S = PL.sample(n=5000, replace = False)
g = sb.PairGrid(data = PL_S, vars = num_variabls)
g = g.map_diag(plot.hist, bins = 20)
g.map_offdiag(plot.scatter);
```



> The scatter plot also demonstrates that BorroerAPR and LoanOrignalAmount are negatively correlated with a -0.3 correlation coefficient, the lower the APR, the greater the loan amount.

In [146…

```
# scatter plot of LoanOriginalAmount vs. BorrowerAPR,

plot.figure(figsize = [8, 6])
plot.scatter(data = PL_S, x = 'LoanOriginalAmount', y = 'BorrowerAPR', alpha = 0.05, col
plot.xlabel('LoanOriginalAmount')
plot.ylabel('BorrowerAPR (%)')
plot.show()
```

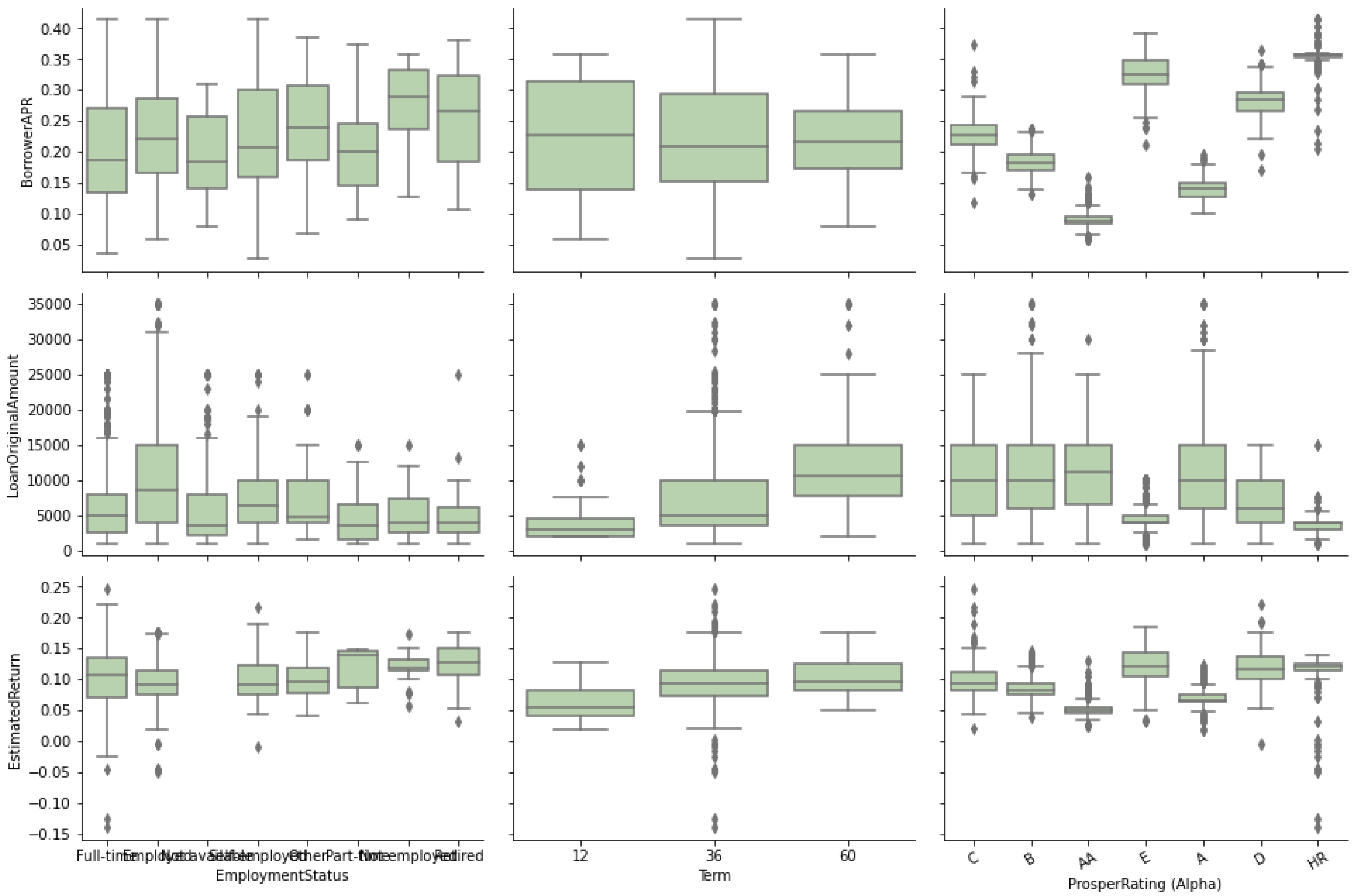


> APR has a wide range at various loan amounts, but that as loan amounts rise, the APR range reduces

In [29]:
```
# plot matrix of numeric features against categorical features.

def boxgrid(x, y, **kwargs):
    sb.boxplot(x, y, color = '#b6d7a8')
plot.figure(figsize = [10, 10])
g = sb.PairGrid(data = PL_S, y_vars = ['BorrowerAPR', 'LoanOriginalAmount','EstimatedRet
                x_vars = cat_variabls, height = 3, aspect = 1.5)
g.map(boxgrid);
plot.xticks(rotation=30);
```

```
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
```

```
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
D:\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follow
ing variables as keyword args: x, y. From version 0.12, the only valid positional argume
nt will be `data`, and passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  warnings.warn(
<Figure size 720x720 with 0 Axes>
```

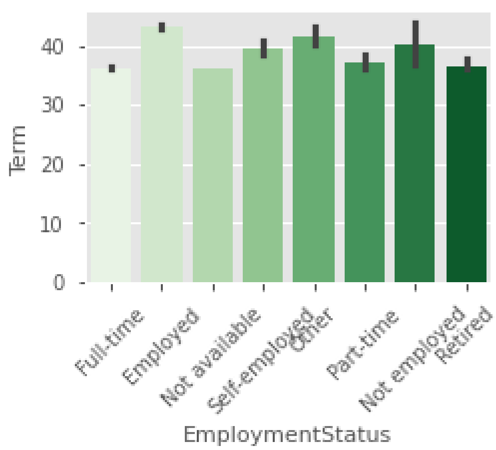


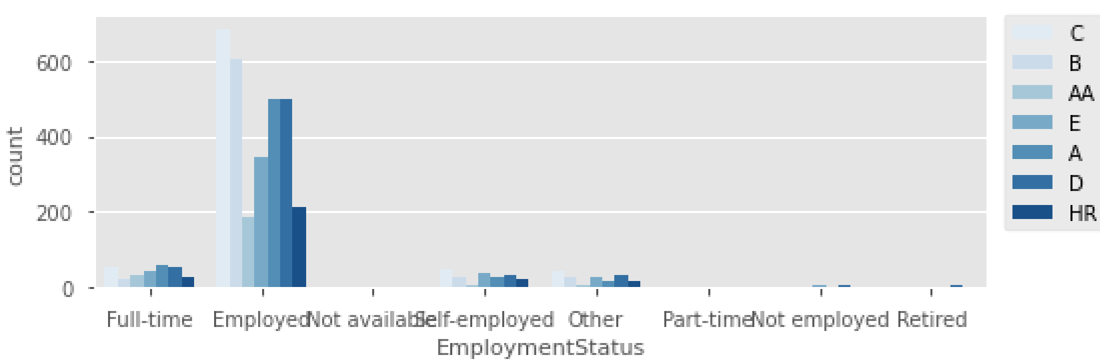
> The lowest APRs are offered by borrowers with the highest Prosper ratings,the loan amount rises as the loan term lengthens, and the better the rating, the lower the borrower APR.

In [144…

```
plot.figure(figsize = [8, 8])

ordered_alpha = ['AA','A','B','C','D','E', 'HR']
# subplot 1: Prosper rating vs term
plot.subplot(2, 2, 1)
sb.barplot(data = PL_S, x = 'ProsperRating (Alpha)', y = 'Term', order = ordered_alpha ,
plot.show()

# subplot 2: employment status vs. term, use different color palette
```

```
plot.subplot(2, 2, 1)
sb.barplot(data= PL_S, x='EmploymentStatus', y='Term', palette = 'Greens')
plot.xticks(rotation = 45)
plot.show()

# subplot 3: Prosper rating vs. employment status
plot.subplot(2, 1, 1)
aix=sb.countplot(data = PL_S, x = 'EmploymentStatus', hue = 'ProsperRating (Alpha)', pal
plot.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)
plot.show()
```





> It is clear that term and Prosper rating interact in some way, there are proportionally more 60-month loans with B and C grades. Borrowers with HR ratings can only get loans for 36 months.

Talk about some of the relationships you observed in this part of the investigation. How did the feature(s) of interest vary with other features in the dataset?

> The scatter plot also demonstrates that BorroerAPR and LoanOrignalAmount are negatively correlated with a -0.3 correlation coefficient, the lower the APR, the greater the loan amount.

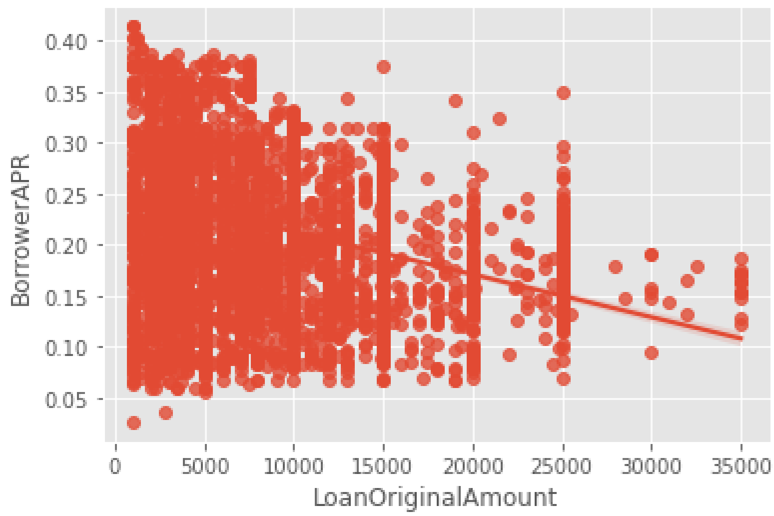
Did you observe any interesting relationships between the other features (not the main feature(s) of interest)?

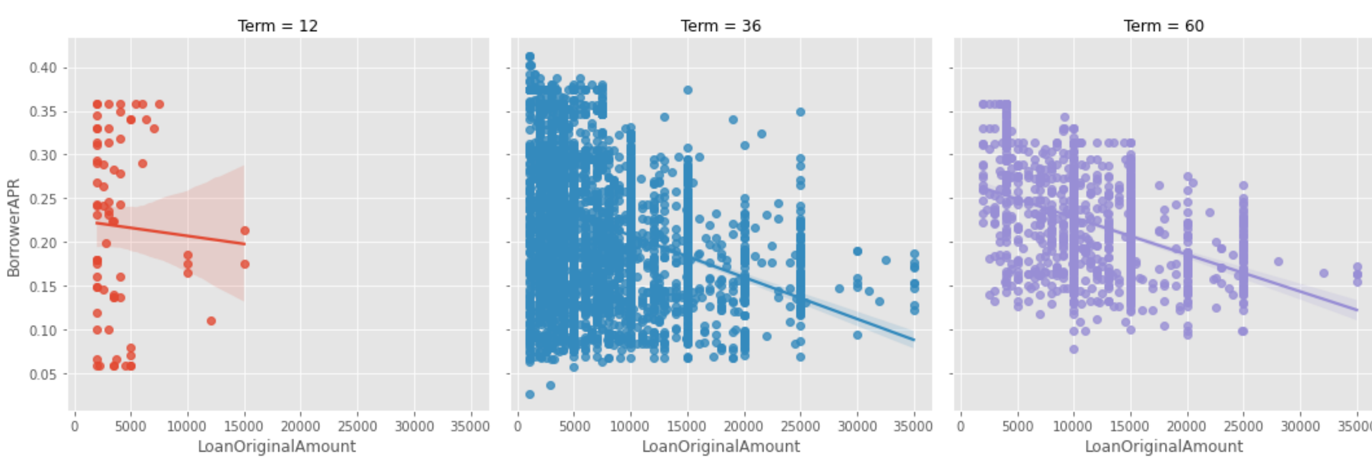
> It is clear that term and Prosper rating interact in some way, there are proportionally more 60-month loans with B and C grades.

Multivariate Exploration

In this section of the analysis, my main focus is on how the categorical variables—Prosper rating and term—affect the correlation between borrower APR and loan orignal amount.

In [55]:
```
# term effects on the APR/loan amount relationship
sb.regplot( x='LoanOriginalAmount',  y='BorrowerAPR', data=PL_S)
plot.xlabel( 'LoanOriginalAmount')
plot.ylabel('BorrowerAPR');
sb.lmplot(x = 'LoanOriginalAmount', y = 'BorrowerAPR', data = PL_S, hue = 'Term', col =
```

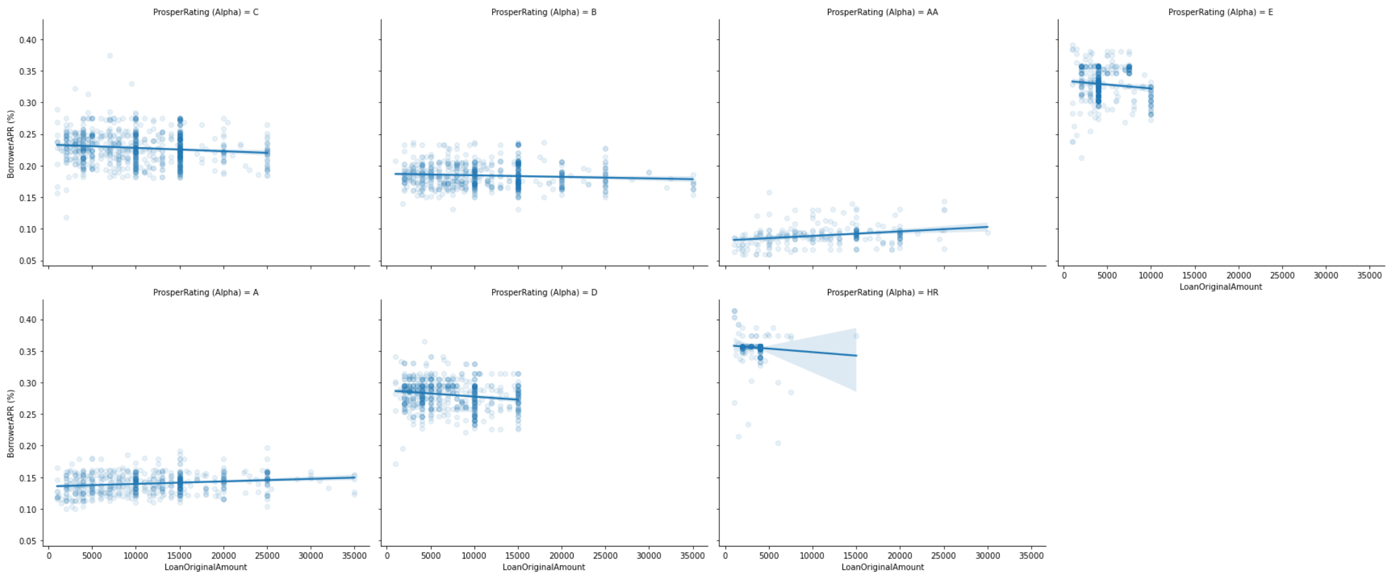




> The relation between the loan amount and APR appears to be unaffected by the term.

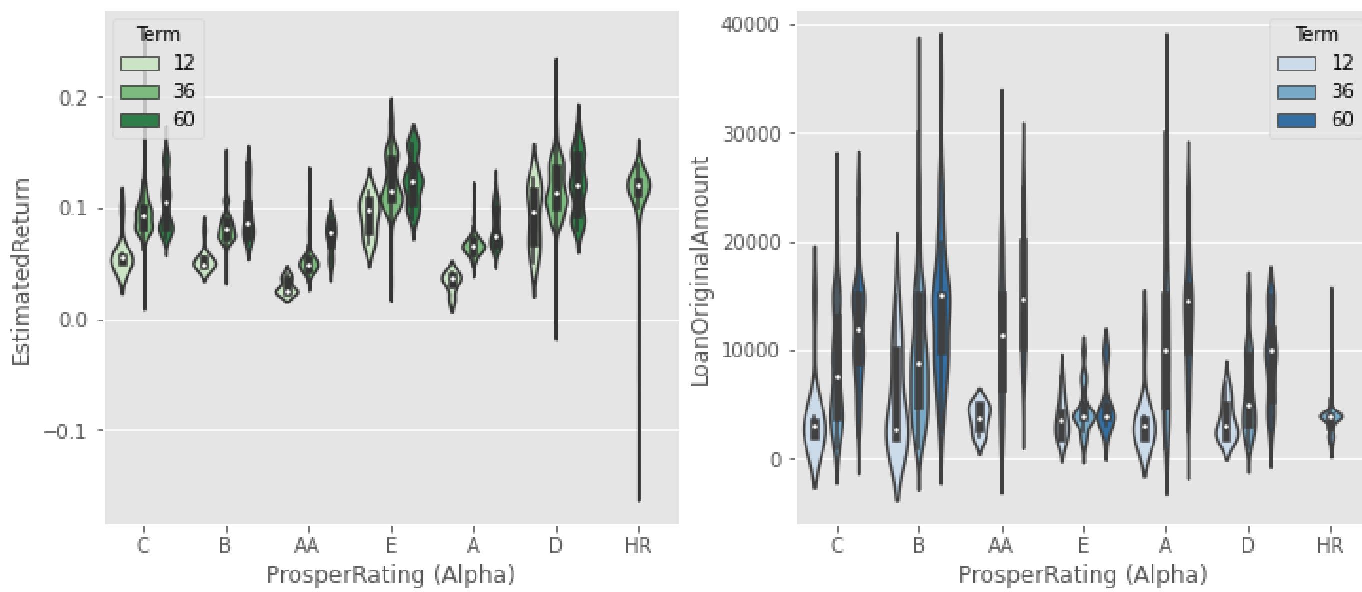
In [42]:
```
# Prosper rating effect on APR and loan amount relationship
g=sb.FacetGrid(data=PL_S, aspect=1.2, height=5, col='ProsperRating (Alpha)', col_wrap=4)
g.map(sb.regplot, 'LoanOriginalAmount', 'BorrowerAPR', x_jitter=0.04, scatter_kws={'alph
g.set_xlabels('LoanOriginalAmount')
```

```
g.set_ylabels('BorrowerAPR (%)')

plot.show()
```



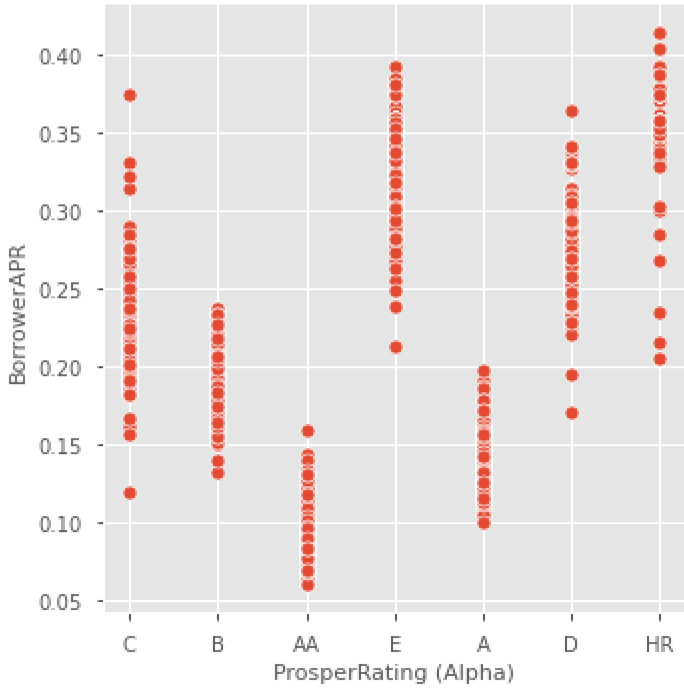
> A higher rating raises the loan amount. A better rating lowers the borrower APR.

In [65]:
```
fig, ax = plot.subplots(ncols=2, figsize=[12,5])
sb.violinplot(data = PL_S, x = 'ProsperRating (Alpha)', y = 'EstimatedReturn', hue = 'Te
            palette = 'Greens', linestyles = '', dodge = 0.4, ax=ax[0])
sb.violinplot(data = PL_S, x = 'ProsperRating (Alpha)', y = 'LoanOriginalAmount', hue =
            palette = 'Blues', linestyles = '', dodge = 0.4, ax=ax[1]);
```



> There is a relationship between term and Estimated Return for loan amount. We can see that a higher Prosper rating results in higher loan amounts, the same goes for LoanOriginalAmount.

In [85]:
```
plot.style.use('seaborn-notebook')
sb.relplot(x = 'ProsperRating (Alpha)', y = 'BorrowerAPR', data = PL_S );
plot.show()
```



> It's interesting to note that for borrowers with HR-C rates, the borrower APR decreases as the borrow period lengthens. However, the APR rises as the length of the loan increases for borrowers with B-AA grades.

Talk about some of the relationships you observed in this part of the investigation. Were there features that strengthened each other in terms of looking at your feature(s) of interest?

> The results of the multivariate analysis revealed that when the Prosper ratings rise from HR to AA, the link between borrower APR and loan amount shifts from being negatively to sluggishly positively. I then looked into how terms and ratings affected loan amounts, and the results showed that with better Prosper ratings, the loan amounts for all three terms increased

Were there any interesting or surprising interactions between features?

> Unexpectedly, the borrower APR and loan amount have a negative link when the borrower's Prosper rating is between HR and B, but a positive correlation when the borrower's rating is between A and AA. Another intriguing finding is that for borrowers with HR-C rates, the borrower APR decreases as the borrow time lengthens. However, the APR rises with the length of the loan for those with B-AA credit ratings.