Part II - Impact of Loan Features on a Borrower

by Muyul Alsubaie

Investigation Overview

> I intended to look at the features of loans that could be utilized to estimate their borrower APR in this analysis. The original loan amount, borrower's APR, were the key points of emphasis.

Dataset Overview

> 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.

In [2]:
```
# 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

# suppress warnings from final output
import warnings
warnings.simplefilter("ignore")
```

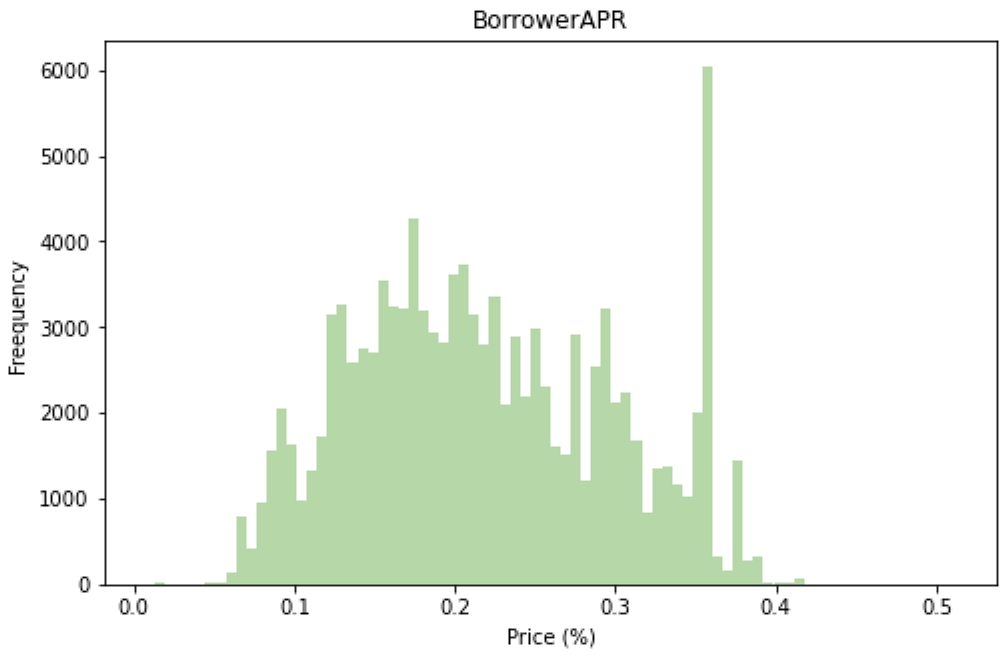
In [3]:
```
# load in the dataset into a pandas dataframe
PL = pd.read_csv('prosperLoanData.csv')
```

Distribution of Borrower APR

> A multimodal distribution of APR appears. 0.1 for the tiny peak and 0.2 for the huge peak. The APR for most loans is 0.43 or less.

In [4]:
```
#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 [5]:
```
#bins = np.arange(0, PL['BorrowerAPR'].max()+binsize, binsize)
display_plot( PL['BorrowerAPR'], 'Price (%)', 'BorrowerAPR')
plot.xlabel('Price (%)')
plot.ylabel('Freeqency ')
plot.title('BorrowerAPR');
```

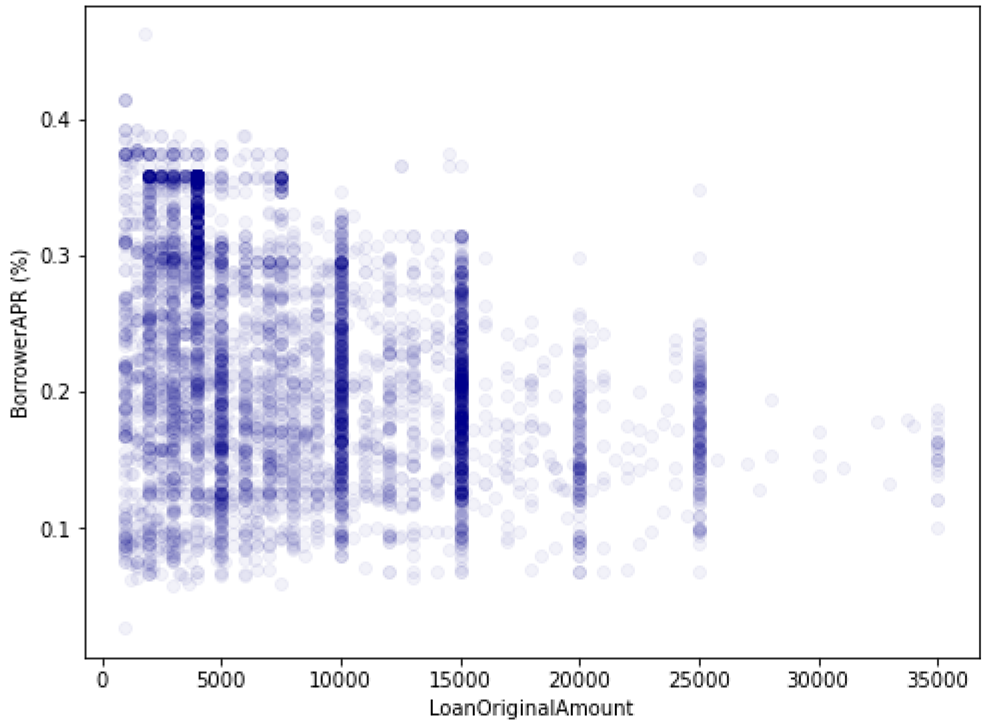


In [7]:
```
num_variabls = [ 'BorrowerAPR', 'LoanOriginalAmount','EstimatedReturn','DebtToIncomeRati
cat_variabls  = ['EmploymentStatus','Term', 'ProsperRating (Alpha)']
PL_S = PL.sample(n=5000, replace = False)
```

Borrower APR vs. Loan Amount

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

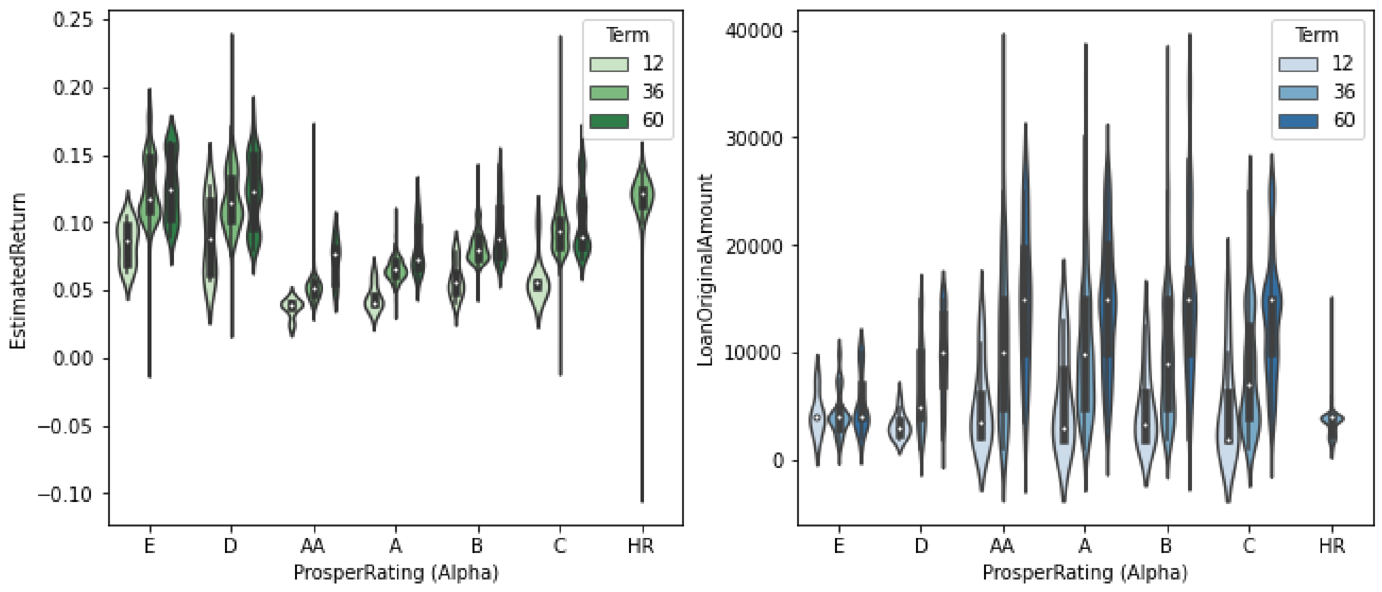
In [8]:
```
# scatter plot of LoanOriginalAmount vs. BorrowerAPR,

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()
```



Interaction between term and Estimated Return for loan amount

> > 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 [10]:
```
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]);
```



> **Generate Slideshow**: Once you're ready to generate your slideshow, use the `jupyter nbconvert` command to generate the HTML slide show. . From the terminal or command line, use the following expression.

In [19]:
```
!jupyter nbconvert <Part_II_Filename>.ipynb --to slides --post serve --no-input --no-pro
```

```
The system cannot find the file specified.
```

In [ ]: