Lendingclub.com loan data

Exploring loans from 2007 to 2015 that have either been completed or ‘charged off’. We will apply machine learning tools to predict which loans will be completed and which will not.

**Introduction:**

Banks make a significant portion of their money in the form of interest from loans.  In addition to many of the other risks that banks face, an important risk factor is if a loan will be paid off.  The risk that the payer of the loan is unable to pay the full amount of the loan. Thus the ability to foresee who these customers are is valuable so that we can charge them appropriately in order to adequately account for the risk, or deny them completely.

The client could be any variety of firms that hand out loans to potential customers from large banks to small businesses like car dealerships and medical companies looking to keep loans ‘in-house’.  Clients will be able to use this information for a variety of purposes from pricing interest rates to marketing more aggressively to particular audiences.

**Origin of the Data:**

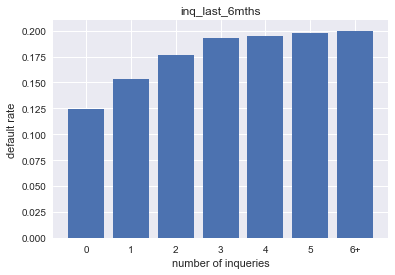
LendingClub is a website that allows individuals and institutions to either apply for loans or sell loans.  The data acquired as a result of these loans from 2007 to present is publicly available here: <https://www.lendingclub.com/info/download-data.action>.  Since some of the data is split up into multiple sheets, we will have to recombine these files to apply algorithms to all these data points.

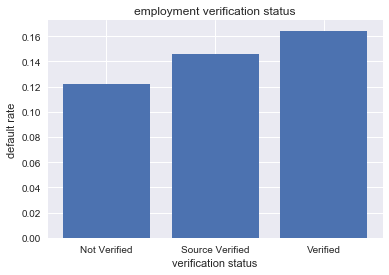
**Data Wrangling:**

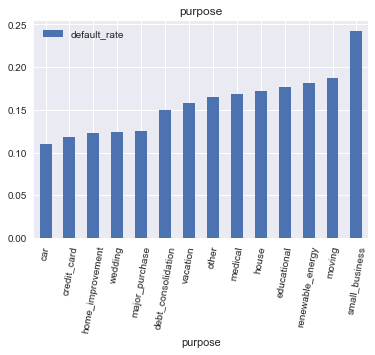
Once the data was loaded after downloading the csv files. We wanted to separate out the loans that had already been completed. There were some loans that were extended for various reasons, but they did not make up a substantial portion of the total loans and were dropped. For purposes of this project, we will consider those whom have been charged off to have defaulted. Columns with dates were not in the pandas datetime format and had to be converted. A new column that counted the length of the borrower’s credit in years was created from the dates. The data set came with up to 151 columns per file, many of which were completely blank or referred to factors that happened after a loan was charged off. The number of columns was reduced to those that had completed or nearly completed columns and used information that was collected before the loan began. This reduced the columns down to 26. Next, percentage signs were removed and restated as a decimal. When setting up for modeling, features that used string statements were converted to binary variables using the pandas.get\_dummies function. Three more binary variables were created to distinguish borrowers that: had a dti ratio over sixteen, had an income over $50,000 per year, and had a revolving utility above fifty percent. Before we begin testing, we must make the data balanced and we will accomplish this through up sampling the data. Many of the other parameters have varying scales and are normalized so that one feature does not overwhelm the models.

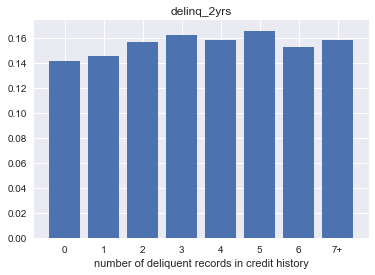
**Exploratory Data Analysis:**

After reducing the loans to just those that had been completed, we were still left with $5.6 billion in loans with an average loan of about $13,000. Inquiries in the last six months and delinquencies in the last two years had long narrow tails that were binned together and plotted as their respective own columns. The variables: pub\_rec, revol\_util, revol\_bal, dti, total\_acc also had outliers, but these were with the other histograms. Longer loans tend to default more often than shorter loans. The more inquiries into a borrower’s credit, the more likely they are to not pay off the loan. Borrowers that verified their employment status, were surprisingly slightly more likely to default than those who did not. Small business loans were more likely to be written off. Delinquent payments in the last two years did not have a trend up or down to predict defaults. Those with credit lengths longer than ten years were less likely to default than those with credit lengths less than ten years. DTI ratios less than nineteen were preferable to those with ratios above nineteen. Clients that made over $50,000 were more likely to pay off their loans than those who made less. Those who used less than fifty percent of their available credit were more likely to pay off their loans.

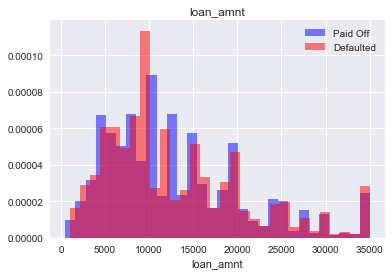


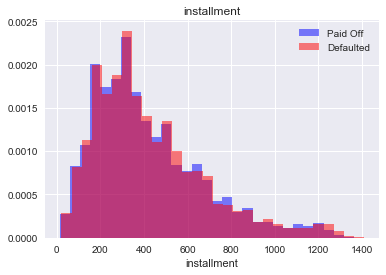
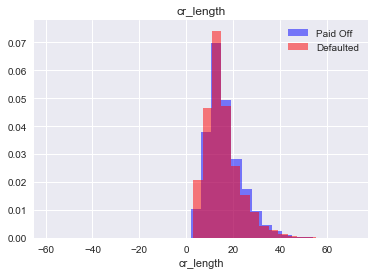


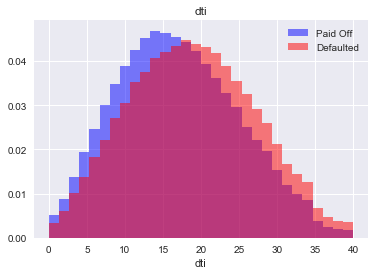
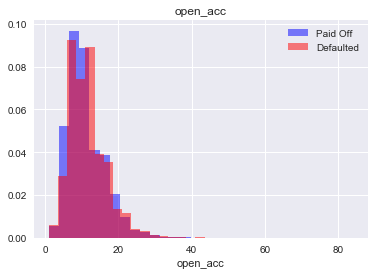


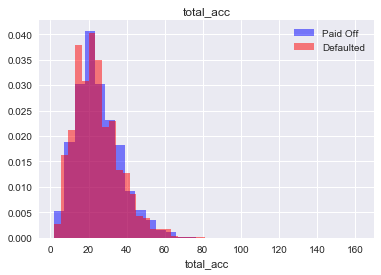
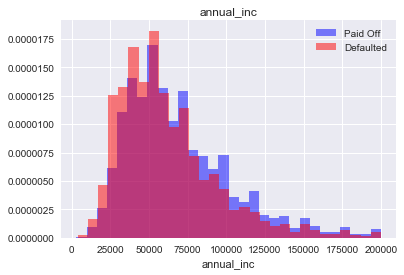


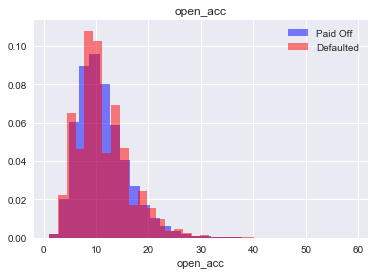
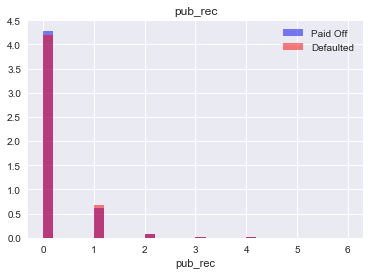
Normalized Histograms to display differences in data:

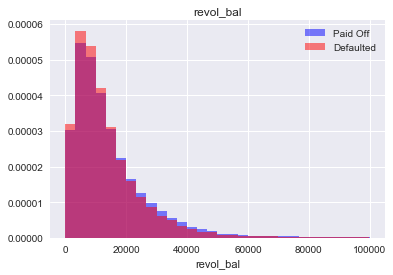
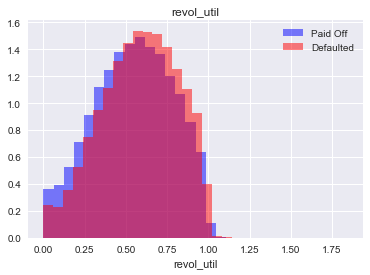












As one can see, most of these variables overlap quite a bit and it is difficult to extract meaningful differences in the data between those that default versus those who pay back their loans in full. This makes this problem a great set up for the application of machine learning algorithms. We will try a variety of approaches and see which ones preform the best.

**Applying models:**

Four different classifier models were tried to predict how many loans would be fully paid versus how many would not. Precision will be important for targeting customers that the investor should either spend more resources acquiring. Recall explains how many customers will be turned away based on the model that would have otherwise fully paid their loan. All three models produced a precision for fully paid off loans above 0.87, but the recall score varied between 0.58 and 0.87. Despite it’s reputation for running slow, the Random Forest model completed the fastest. The model with the highest F1-score was the K-Nearest Neighbors at 0.87, however it took almost three and a half hours to fit to the data and was the fastest in predicting the test set. Moving forward, I would recommend the K Nearest Neighbors model as it has the best balance between precision and recall. If an investor had limited capital or just wanted to focus on simply reducing as many defaults as possible, the logistic regression would be the best fit with a precision of 0.92.

Metrics for classifying paid off loans.

