Proposal

March 26, 2018

1 Machine Learning Engineer Nanodegree

1.1 Capstone Proposal - Predicting Write-offs in LendingClub Loan Data Set

Antal Berenyi March 25, 2018

1.2 Proposal

1.2.1 Domain Background

Lending Club (LC) operates an online, peer-to-peer lending platform that enables borrowers to obtain a loan, and investors to purchase notes backed by payments made on loans. The company claims that $15.98 billion in loans had been originated through its platform up to December 31, 2015. wikipedia Loans can be issued for a variety of purposes, such as loan consolidation, car purchase, medical, etc. Loans are issued as $25 notes so that lenders may diversify their investment over many loans. Loan terms are either 36 or 60 months.

LC divides loans into categores A,B,C,D,E,FG, based on variables that measure the quality of the loan. A is highest and FG is lowest. Category A loans offers the lowest rate of return backed by borrowes with highest credit rating therefore it is the safest investment. FG are the riskiest loans with high interest rate but least likely to be repaid.

source

If a borrower does not pay a loan installment on time, its status changes from `current` to `grace period` for 14 days. After that the loan status changes to `delinquent`, then to `late`, then `charged off` after 3 months if not paid.

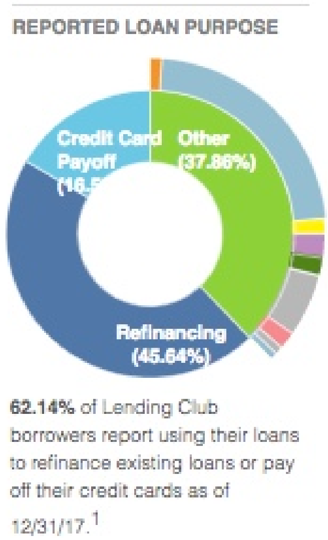
I have been investing in LC for about 3 years. Over this time period I have invested in a range of loan classes. About 50% of the interest earned was erased by load charge-offs. Identifying loans likely to be charged-off means that investor profit could be increased by not investing in those notes. invesing in LC

1.2.2 Problem Statement

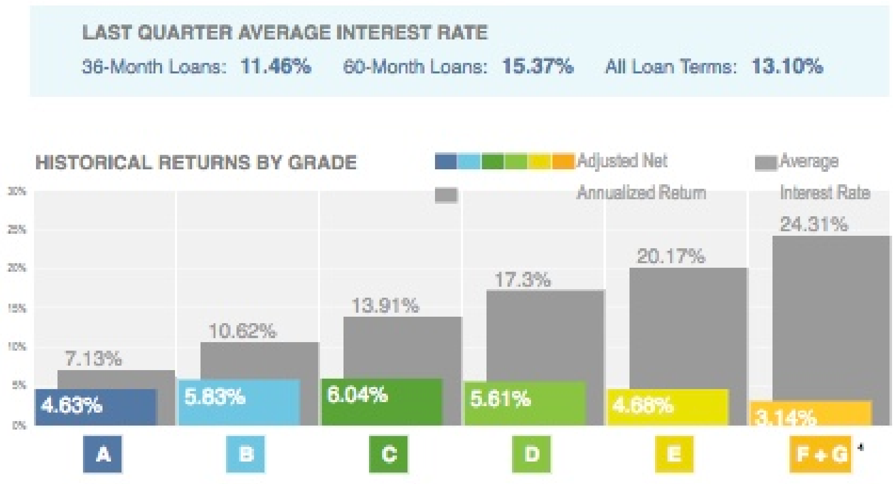
The problem to be solved is to identify which loans are likely to be charge-off. Charge-offs impact investor returns because investors lose both investment capital plus the potential to earn interest.

Hypothetical Projected Net Return Example[1]:		
	Average interest rate for portfolio[2]	14%
	Estimated effect of charge-offs and prepayments[3]	-8%
	Effect of LendingClub fees[4]	-1%
	Annualized Net Return[5]	**5%**

source



loan purpose



interest%20rate.jpeg

To solve this problem we need to identify a correlation between loan features and "charge-off" probability. I am planning to train a classifier on the data set to classify loans as "charge-off" vs. "non-charge-off".

To quantify the problem: the charge-off rate can be calculated as the percentage of loans with status "charge-off" from the data set.

To measure performance: The trained classifier should perform better than a naive classifier; it should identify charge-off loans to a greater accuracy than a naive classifier. The naive classifier picks notes randomly with the same probability as the percentage of actual `charge-off` loans.

To replicate the results: The classifier then can be tested for validity on a testing data set. LC publishes the list of loans rejected and this data set can be a good cross-reference for the validity of the classifier, althought the exact criteria LC uses are not known.

1.2.3 Datasets and Inputs

LC collects extensive statistics on borrowers that they make their rating based on. The borrower data used for this project is publicly available on the LC web site download-data. It contains all the LendingClub loan information available for investors to make a decision about whenter to fund the loan or not. The LCDataDictionary.xlsx file lists the feature with explanation about each feature. The data is in zipped up .csv format that can be imported in Excel, Pandas or other tools.

This data set should contain a mix of 3yr and 5yr loans that originated up to 5 years ago and new newer loans, with all possible status. For example the Q1 2017 data set contains anonymized information for about 96,781 loans with 151 features.

The zipped file size is about 22MB while unzipped it is 110 MB. Each row represents a loan. Each column is a feature. Column "loan_status" is the target variable to be predicted. The finacial data set is donwloadable in quarterly and yearly chunks. These features relate information about the loand and the borrower: `loan amount, interest rate, grade, purpose; borrower income, geography, employment status, FICO score, etc.` This information will be used to find a good indicator (predictor) of why a borrower may default on a loan.

```
In [3]: import pandas as pd
        df = pd.read_csv('https://resources.lendingclub.com/LoanStats_2017Q1.csv.zip',
                         skiprows=1, compression='zip', low_memory=False)
        display(df.head())
```

```
    id  member_id  loan_amnt  funded_amnt  funded_amnt_inv        term  \
0  NaN        NaN    15000.0      15000.0          15000.0   36 months
1  NaN        NaN    17000.0      17000.0          17000.0   36 months
2  NaN        NaN    20000.0      20000.0          20000.0   36 months
3  NaN        NaN    16000.0      16000.0          16000.0   60 months
4  NaN        NaN     2000.0       2000.0           2000.0   36 months

  int_rate  installment grade sub_grade        ...          \
0    5.32%       451.73     A        A1        ...
1    7.49%       528.73     A        A4        ...
2    5.32%       602.30     A        A1        ...
3   12.74%       361.93     C        C1        ...
4   16.99%        71.30     D        D1        ...
```

```
  hardship_payoff_balance_amount hardship_last_payment_amount  \
0                            NaN                          NaN
1                            NaN                          NaN
2                            NaN                          NaN
3                            NaN                          NaN
4                            NaN                          NaN

  disbursement_method  debt_settlement_flag debt_settlement_flag_date  \
0                Cash                     N                       NaN
1                Cash                     N                       NaN
2                Cash                     N                       NaN
3                Cash                     N                       NaN
4                Cash                     N                       NaN

  settlement_status settlement_date settlement_amount  settlement_percentage  \
0               NaN             NaN               NaN                    NaN
1               NaN             NaN               NaN                    NaN
2               NaN             NaN               NaN                    NaN
3               NaN             NaN               NaN                    NaN
4               NaN             NaN               NaN                    NaN

   settlement_term
0              NaN
1              NaN
2              NaN
3              NaN
4              NaN

[5 rows x 145 columns]
```

In [4]:
```
pd.options.display.max_rows = 20
pd.read_excel('https://resources.lendingclub.com/LCDataDictionary.xlsx')
```

Out[4]:
```
                    LoanStatNew  \
0                acc_now_delinq
1          acc_open_past_24mths
2                    addr_state
3                      all_util
4                    annual_inc
5              annual_inc_joint
6              application_type
7                   avg_cur_bal
8                bc_open_to_buy
9                       bc_util
..                          ...
143         disbursement_method
144        debt_settlement_flag
```

```
145  debt_settlement_flag_date
146           settlement_status
147             settlement_date
148           settlement_amount
149       settlement_percentage
150             settlement_term
151                         NaN
152                         NaN

                                                   Description
0    The number of accounts on which the borrower i...
1           Number of trades opened in past 24 months.
2    The state provided by the borrower in the loan...
3                Balance to credit limit on all trades
4    The self-reported annual income provided by th...
5    The combined self-reported annual income provi...
6    Indicates whether the loan is an individual ap...
7              Average current balance of all accounts
8            Total open to buy on revolving bankcards.
9    Ratio of total current balance to high credit/...
..                                                 ...
143  The method by which the borrower receives thei...
144  Flags whether or not the borrower, who has cha...
145  The most recent date that the Debt_Settlement_...
146  The status of the borrowers settlement plan.  ...
147  The date that the borrower agrees to the settl...
148  The loan amount that the borrower has agreed t...
149  The settlement amount as a percentage of the p...
150  The number of months that the borrower will be...
151                                                NaN
152  * Employer Title replaces Employer Name for al...

[153 rows x 2 columns]
```

In [6]:
```python
print("Loan status percentages:")
df[['loan_status']].groupby('loan_status').size() / len(df) * 100
```

```
Loan status percentages:
```

Out[6]:
```
loan_status
Charged Off            3.345698
Current               76.620411
Default                0.005166
Fully Paid            16.581767
In Grace Period        0.989864
Late (16-30 days)      0.406071
Late (31-120 days)     2.048956
dtype: float64
```

1.2.4 Solution Statement

Data exploration The LC data set contains both loan and borrower features that can be indicative of the evetual charge-off of the loan. Feature values indicating charge-off within each class A,B,C, etc could be different from each other so it is worth-while investigating each class. In the preliminary data exploration I will investigate each feature. I will calculate statistics to determine if they need to be normalized, encoded, missing values supplied, appropriate imputation strategy selected. Variable distributions will be investigated and if appropriate features will be scaled, combined or omitted.

Model I am planing to compare DecisionTree, SVM, NeuralNet, KNN and other classifier performances. It may be useful to investigate dimensionality reduction to reduce the number of input dimensions by Principal Component Analysis and clustering. I will select the best parameters for the classifier using GridSearch and a scoring function using recall_score.

Performance Metric I am planning to measure performance by Recall. The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples (charge-offs). This problem is similar to a fraud detector where we aim to catch all positive cases. We don't want a very high rate of False positives either because that would eliminate valid investment opportunities, so reporting a Precision score is also meaningful.

Evaluating performance To measure performance: The trained classifier should perform better than a naive classifier; it should identify charge-off loans with a better Recall than a naive classifier. The naive classifier picks notes randomly with the same probability as the percentage of actual `charge-off` loans.

```
In [7]: from sklearn.metrics import precision_recall_fscore_support
        from sklearn.metrics import accuracy_score
        from sklearn.metrics import f1_score

        c_o = 334
        total = 10000
        ok = total-c_o
        y_true = [1]*c_o + [0]*ok

        #all_charge-off
        y_pred = [1] * total
        print("All charge-off model:")
        a = accuracy_score(y_true, y_pred)
        p,r,f2,s = precision_recall_fscore_support(y_true, y_pred, warn_for=(), beta=2, average
        f1 = f1_score(y_true, y_pred, average='binary')
        print("accuracy: {}; precision: {}; recall: {}; f1: {:.4f}; f2: {:.4f}".format(a,p,r,f1

        #all_OK
        y_pred = [0] * total
        print("All OK model:")
        a = accuracy_score(y_true, y_pred)
```

```
p,r,f2,s = precision_recall_fscore_support(y_true, y_pred, warn_for=(), beta=2, average
f1 = f1_score(y_true, y_pred, average='binary')
print("accuracy: {}; precision: {}; recall: {}; f1: {:.4f}; f2: {:.4f}".format(a,p,r,f1

#Naive predictor
y_pred = [1]*(c_o//2 ) + [0]*(c_o//2 ) + [1]*(ok//2 ) + [0]*(ok//2 )
print("Naive predictor:")
a = accuracy_score(y_true, y_pred)
p,r,f2,s = precision_recall_fscore_support(y_true, y_pred, warn_for=(), beta=2, average
f1 = f1_score(y_true, y_pred, average='binary')
print("accuracy: {}; precision: {}; recall: {}; f1: {:.4f}; f2: {:.4f}".format(a,p,r,f1
```

```
All charge-off model:
accuracy: 0.0334; precision: 0.0334; recall: 1.0; f1: 0.0646; f2: 0.1473
All OK model:
accuracy: 0.9666; precision: 0.0; recall: 0.0; f1: 0.0000; f2: 0.0000
Naive predictor:
accuracy: 0.5; precision: 0.0334; recall: 0.5; f1: 0.0626; f2: 0.1318
```

```
/Users/aberenyi/anaconda2/envs/Py36/lib/python3.6/site-packages/sklearn/metrics/classification
  'precision', 'predicted', average, warn_for)
```

1.2.5 Benchmark Model

The benchmark model I intend to use is a naive predictor. It will randomly classify a loan as `charge-off` with the probability of occurence of charge-off loans in the whole data set.

Define the percentage of charged off loans calculated from the data set as P0. That is estimated about 3.335% A naive predictor is built to randomly label a data point as "charge-off" with probability PN = 0.5. Calculate the Recall score of the naive predictor. This will be the benchmark against which the solution will be beasured.

For a model that predicts all notes as charge-off, the metrics would be:

Accuracy	Precision	Recall	F1	F2
0.0335	0.0335	1	0.0648	0.148

For a model that predicts all notes as OK, the metrics would be:

Accuracy	Precision	Recall	F1	F2
0.9665	0	0	0	0

For the Naive predictor, given about 3.35% charge-off rate, the metrics would be:

Accuracy	Precision	Recall	F1	F2
0.5	0.0335	0.5	0.0627	0.132

Clearly, a better model would have a better Recall than 50% and F2 score greater than 0.132 classification-accuracy-is-not-enough

1.2.6 Evaluation Metrics

The best evaluation metric would be Recall. We want to label all charged off loans to minimize our loss. It is OK to mislabel a few loans as false positive, so the situation is similar to a fraud detector or spam filter, where we are aiming for high Recall. Other metrics to better than Naive predictor: Accuracy > 0.5 and F2 > 0.132. We would like to keep the accuray as high as possible, close to 0.9665.

1.2.7 Project Design

The workflow shall be as follows: > 1. Select from the available data sets one or more that contain enough sample data. Read in each and take note of the ratio of charge-off to good notes.

2. Read in the data set and perform preliminary data analysis. Inspect each feature, evaluate statistics, fill in missing values and encode categorical data. Eliminate columns if they are not likely to be useful. Data may need to be normalized or log normalized based on distribution.

3. Split data into feature-target. Target will be 'loan_status' column. Binary encode target column.

4. Split the data into traing, validation and testing sets.

5. Perform dimensionality reduction if possible. I plan to perform Linear Discriminant Analysis to reduce the dimensionality of the input by projecting it to the most discriminative directions.

6. Analyze model performance for several classifiers using training and validation sets. Measure the metrics of each classifier and compare them against the benchmark and each other. The model with the best metrics and potential for improvement will be tuned with GridSearch. Traing curves will be used to avoid overfitting the data and gauge whether there is enough data for the algorithm.

7. Report model performance on testing set.

8. Predict charge-off loans using model to mark for removal from portfolio.

I consider the following classification algorithms in my implementation: SVM, DecisionTree, KNN, Gaussian Mixture Model. Neural Networks are a good candidate solution, trying different achitectures (2 layer?)

*Zip codes can be encoded as leave as is or frequency.