Lending Club

HarvardX - PH125.9x Data Science Capstone

Emmanuel Rialland - https://github.com/Emmanuel_R8

October 31, 2019

Contents

List of Tables

List of Figures

Introduction

Lending Club (*LC*) is an American company listed on the New York stock exchange that provides a platform for peer-to-peer lending. Unlike banks, it does not take deposits and invest them. It is purely a matching system. Each loan is split into $25 that multiple investors can invest in. LC is remunerated by fees received from both sides. LC states that they have intermediated more than $50bln since they started operations. Further description of the company is easily available online numerous sources.

In order for investors to make the best investment decisions, LC make a historical dataset publicly available to all registered investors. This dataset is the subject of this report. It was downloaded from the Kaggle data science website[1].

The size of the dataset is rich enough that it could be used to answer many different questions. We decided for a focused approach. Following Chapter 5 of Peng (2012), we will first formulate the question we want to answer to guide our analysis.

The business model of LC is to match borrowers and investors. Naturally, more people want to receive money than part with it. An important limiting factor to LC's growth is the ability to attract investors, build a trusting relationship where, as a minimum first step, investors trust LC to provide accurate, transparent and reliable information of the borrowers. For this purpose, LC decided not only to provide extensive information about potential borrowers' profile, but also historical information about past borrowers' performance. This is, as we understand, one of the key purposes of this dataset. We decided to use the dataset for this very purpose. Essentially, the questions are: **given a borrower profile, is his/her rating appropriate in terms of risk of default? And if a default occurs, what is the expected recovery? The summary question is: given a borrower profile, is the risk/reward balance appropriate to commit funds?** In answering this question, we understand that LC allows investment of very granular amounts. Therefore, even an individual investor can diversify his/her loan and risk portfolio. It is not necessary to 'gamble' funds on a single borrower. This is exactly what institutional investors achieve through syndication (although on a very different scale, typically $10-25mln for a medium-size bank).

For this exercise, we made two simplifying (hopefully not simplistic) assumptions:

- In determining the risk/return balance, we have not accounted for LC's cost of intermediation. By ignoring fees paid by both sides, we obviously overestimate the returns to the investors. But in first approximation, **we will assume that the risk/reward balance, from the investors' point of view, across ratings is independent from fees.** This is a simplification. Real-world fees are higher the lower the investment grade and push the investors to receive, and the borrowers to pay, higher interest margin.
- All-in interest rates paid by borrowers are fixed. This is highly desirable for borrowers to be able to manage their cashflow. However, an investor should always consider an investment return as a margin above a risk-free return. Banks would look at LIBOR; bond investors (e.g. life insurers) would look at government bonds. Those risk-free rates can change very quickly, whereas we understand that LC sets those rates on a

[1] https://www.kaggle.com/wendykan/lending-club-loan-data/data

less frequent basis. In other word, the risk premium will vary rapidly. **We assume that individual investors are 'in-elastic' to change in implied risk premia.** But we recognise this as a limitation of our work.

This report is organised as follows:

- [XXXX]

Chapter 1

Dataset

The data is sourced as a *SQLite* database that was imported as a `dataframe` with the `RSQLite` package. The variables were reformatted according to their respective types.

We also sourced US zip and FIPS codes, and macroeconomical data for possible geographical statistics. The source code for the data import and reformatting is given in appendix.

1.1 Preamble

The LendingClub dataset, although rich, is difficult to interpret. The only explanation of what the variables mean comes from a spreadsheet attached to the dataset. The explanations are not precise and/or subject to conflicting interpretation. Despite serching the LendingClub website, no further original information was found. We collected a number of reasonable assumptions in Appendix.

The dataset has been used a number of times in the past by various people. One paper (Kim and Cho, 2019) mentions they used a dataset that included 110 variables, which is less than ours with 145 variables. The dataset has changed over time in ways we do not know.

1.2 General presentation

The original dataset is rich: it includes 2260668 loan samples, each containing 144 variables (after the identification variables filled with null values). The loans were issued from 2007-06-01 to 2018-12-01.

1.2.1 Business volume

The dataset represents a total of ca.$34bln in loan principals, which is a substantial share of the total amount stated to have been intermediated to date by LC (reported to be $50bln+). About 60% of the portfolio is fully repaid. See Table 1.1.

Figure 1.5 plots the number, volume (cumulative principal amount) and average principal per loan. It shows that the business grew exponentially (in the common sense of the word) from inception until 2016. At this point, according to Wikipedia [1]:

" *Like other peer-to-peer lenders including Prosper, Sofi and Khutzpa.com, LendingClub experienced increasing difficulty attracting investors during early 2016. This led the firm to increase the interest rate it charges borrowers on three*

[1] source: https://en.wikipedia.org/wiki/LendingClub - Retrieval date 15 September 2019

Table 1.1: Number of loans per status

Loan status	Count	Proportion (%)
Charged Off	261655	11.574
Current	919695	40.682
Default	31	0.001
Does not meet the credit policy. Status:Charged Off	761	0.034
Does not meet the credit policy. Status:Fully Paid	1988	0.088
Fully Paid	1041952	46.090
In Grace Period	8952	0.396
Late (16-30 days)	3737	0.165
Late (31-120 days)	21897	0.969

occasions during the first months of the year. The increase in interest rates and concerns over the impact of the slowing United States economy caused a large drop in LendingClub's share price."

The number and volume of loans plotted have been aggregated by month. The growth is very smooth in the early years, and suddenly very volatile. As far as the first part of the dataset is concerned, a starting business could expect to be volatile and could witness a yearly cycle (expected from economic consumption figures) superimposed on the growth trend. This is not the case.

An interesting metric is that the average principal of loans has increased (see Figure **??**, on a sample of 100,000 loans). Partly, the increase in the early years could be interpreted success in confidence building. This metric plateau-ed in 2016 and decreased afterwards, but to a much lesser extent than the gross volume metrics. However, it is more volatile than the two previous metrics in the early years.

By the end of the dataset, all metrics have essentially recovered to their 2016 level.

1.2.2 Loan lifecyle and status

In the dataset, less loans are still outstanding than matured or "*charged off*" (term that LC use to mean partially or fully written off, i.e. there are no possibilty for LC and/or the investors to receive further payments). The share of outstanding loans is:

```
## Share of current loans =  42.214 %
```

The dataset describes the life cycle of a loan. In the typical (ideal) case, we understand it to be:

$$\text{Loan is approved} \longrightarrow \text{Full amount funded by investors} \longrightarrow \text{Loan marked as Current} \longrightarrow \text{Fully Paid}$$

In the worst case, it is:

$$\text{Loan is approved} \longrightarrow \text{Full amount funded by investors} \longrightarrow \text{Loan marked as Current} \longrightarrow$$

$$\longrightarrow \text{Grace period (missed payments under 2 weeks)} \longrightarrow \text{Late 15 to 31 days} \longrightarrow$$

$$\longrightarrow \text{Late 31 to 120 days} \longrightarrow \text{Default} \longrightarrow \text{Charged Off}$$

Note that *Default* precedes and is distinct from *Charged Off* [2]. A couple of things could happen to a loan in default:

- LC and the borrower restructure the loan with a new repayment schedule, where the borrower may repay a lesser amount over a longer period; or,
- the claim could be sold to a debt recovery company that would buy the claim from LC/investors. This would be the final payment (if any) received by LC and the investors.

The dataset also describes situations where a borrower negotiated a restructuring of the repayment schedule in case of unexpected hardship (e.g. disaster, sudden unemployment).

Note that this progression of distinguishing default (event in time) and actual financial loss mirrors what banks and rating agencies do> The former is called the *Probability of Default* (PD), the latter *Loss Given Default* (LGD). Ratings change over time (in a process resembling a Markov Chains). LGD show some correlations with ratings. The dataset, although detailed, does not include the full life of each loan to conduct this sort of analysis (change of loan quality over time). This is an important reason why we decided to focus on the loan approval and expected return.

1.2.3 Loan application

Before a loan is approved, the borrower undergoes a review process that assess his/her capacity to repay. This includes:

- employment situation and income, as well whether this income and possibly its source has been independently verified;
- whether the application is made jointly (likely with a partner or a spouse, but there are no details);
- housing situation (owner, owner with current mortgage, rental) and in which county he/she lives (that piece of information is partially anonymised by removing the last 2 digits of the borrower's zipcode);
- the amount sought, its tenor and the purpose of the loan; and,
- what seems to be previous credit history (number of previous deliquencies). The dataset is very confusing in that regard: it is clear that such information relates to before the loan is approved in the case of the joint applicant. In the case of the principal borrower however, the variable descriptions could be read as pre-approval information, or information gathered during the life of the loan. We have assumed that the information related to the principal borrower is also pre-approval. We also used *Sales Supplements* from the LC website[3] that describe some of the information provided to investors. LendingClub also provides a summary description of its approval process in its regulatory filings with the Securities Exchange Commission (LendingClub Corporation , San Francisco, 2019(@).

1.2.4 Interest rates

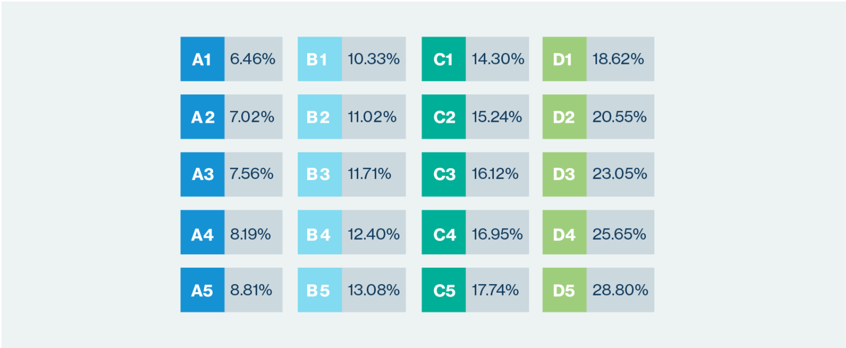
Based on this information, the loan is approved or not. Approval includes the final amount (which could be lower than the amount requested), tenor (3 or 5 years) and a rating similar to those given to corporate borrowers. Unlike corporate borrowers however, the rating mechanically determines the rate of interest according to a grid known to the borrower in advance[4]. The rates have changed over time. Those changes where not as frequent as market conditions (e.g. changes in Federal Reserve Bank's rates)[5].

[2] See LendingClub FAQ at https://help.lendingclub.com/hc/en-us/articles/215488038

[3] See https://www.lendingclub.com/legal/prospectus

[4] https://www.lendingclub.com/investing/investor-education/interest-rates-and-fees

[5] Corporate borrowers would negociate interest margins on a case-by-case basis despite similar risk profiles.



A1	6.46%	B1	10.33%	C1	14.30%	D1	18.62%
A2	7.02%	B2	11.02%	C2	15.24%	D2	20.55%
A3	7.56%	B3	11.71%	C3	16.12%	D3	23.05%
A4	8.19%	B4	12.40%	C4	16.95%	D4	25.65%
A5	8.81%	B5	13.08%	C5	17.74%	D5	28.80%

Figure 1.1: Interest rates given rating

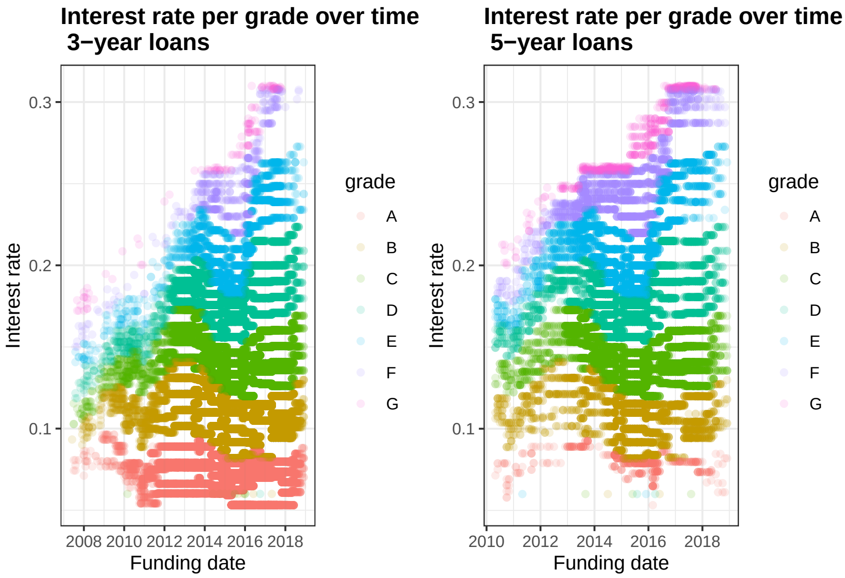


Figure 1.2: Interest rate per grade over time

Figure 1.1 [6] shows the predetermined interest rate depending on the initial rating as of July 2019.

At the date of this report, the ratings range from `A` (the best) down to `D`, each split in 5 sub-ratings. However, LC previously also intermediated loans rated F or G (until 6 November 2017) and E (until 30 June 2019) [7]. This explains that such ratings are in the dataset. We will assume that the ratings in the dataset are the rating at the time of approval and that, even if loans are re-rated by LC, the dataset does not reflect it.

Figures 1.2 shows the change in interest rate over time for different ratings and separated for each tenor. (Each figure is on a sample of 100,000 loans.) For each rating, we can see several parallel lines which correspond to the 5 sub-rating of each rating. We note that the range of interest rates has substantial widened over time. That is, the risk premium necessary to attract potential investors has had to substantially increase. In the most recent years, the highest rates exceed 30% which is higher than many credit cards.3-year loans are, unsurprisingly, considered safer (more A-rated, less G-rated). Identical ratings attract identical rates of interest.

By comparison, we plot the 3-year (in red) and 5-year (in blue) bank swap rates in Figure @(fig:swap-rates). We see that the swap curve has flattened in recent times (3-year and 5-y rates are almost identical). We also can see that in broad terms the interest rates charged reflect those underlying swap rates. It is therefore most relevant to

[6] source: https://www.lendingclub.com/investing/investor-education/interest-rates-and-fees

[7] See https://www.lendingclub.com/info/demand-and-credit-profile.action

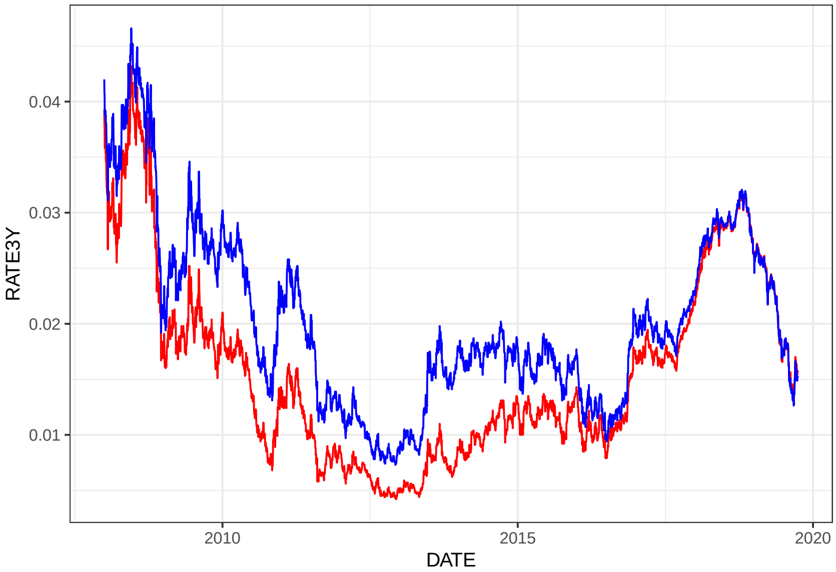


Figure 1.3: Historical Swap Rates

examine the credit margins added to the swap rates.

Figures 1.4 shows the change in credit margin over time for different ratings and separated for each tenor. (Each figure is on a sample of 100,000 loans.) As above, for each rating, we can see several parallel lines which correspond to the 5 sub-rating of each rating. We note that the range of credit margins has widened over time but less than the interest rates. Identical ratings attract identical credit margins.

[TODO: DTI, amount... by grade]

1.2.5 Payments

The loans are approved for only two tenors, 3 and 5 years, with monthly repayments. Installments are calculated easily with the usual formula:

$$Installment = Principal \times \frac{1}{1 - \frac{1}{(1+rate)^N}}$$

Where $Principal$ is the amount borrowed, $rate = \frac{\text{Quoted Interest Rate}}{12}$ is the monthly interest rate, and N is the number of installments (36 or 60 monthly payments). The following piece of code shows that the average error between this formula and the dataset value is about 2 cents. We therefore precisely understand this variable.

```
local({
  installmentError <- loans %>%
    mutate(
      PMT = round(funded_amnt * int_rate / 12 / (1 - 1 / (1 + int_rate / 12) ^
                                                   term), 2),
      PMT_delta = abs(installment - PMT)
    ) %>%
    select(PMT_delta)

  mean(100 * installmentError$PMT_delta)
})
```

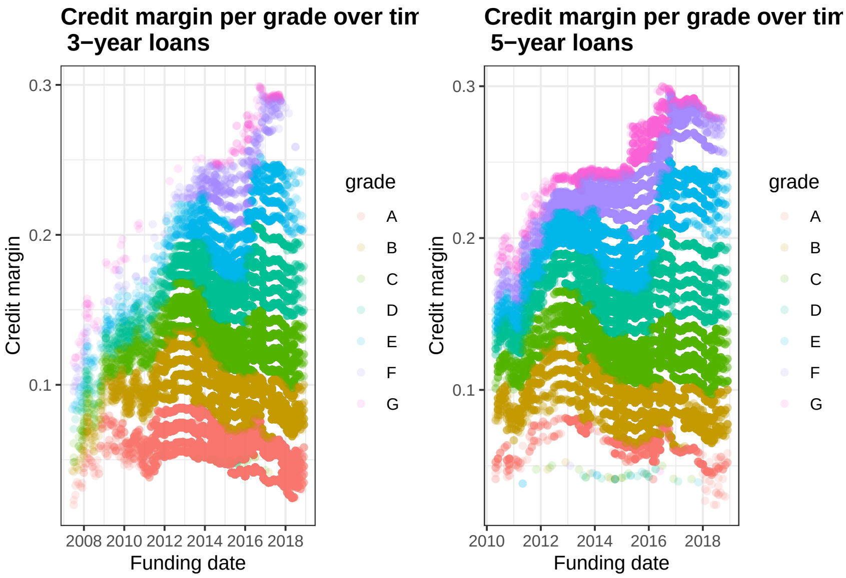


Figure 1.4: Credit margins per grade over time

1.3 Variables

We here present the dataset in a bit more details The full list of variable is given in appendix (see Table 1.3). This dataset will be reduced as we focused on our core question: *Are LC's loans priced appropriately?*.

1.3.1 General

1.3.2 Identification

The dataset is anonymised (all identifying ID numbers are deleted) and we therefore removed those columns from the dataset. Since the identification `IDs` have been removed to anonymise the dataset, we cannot see if a borrower borrowed several times.

1.4 Loan decision

As indicated in the introduction, our focus is on loans that have gone through their entire life cycle to consider their respective pricing, risk and profitability. To that effect, we will remove all loans which are still current (either performing or not). From here on, everything will be based on this reduced dataset.

In this reduced dataset, we focus on loans that have matured or been terminated. It contains 1306356 samples. Most of the loans (ca.80%) have been repaid in full. See Table 1.2.

When grouped by grade (Figure **??**), we see a clear correlation between grade and default: the lower the grade the higher the portion defaults (all the way down to about 50%). In addition, most of the business is written in the B- or C-rating range.

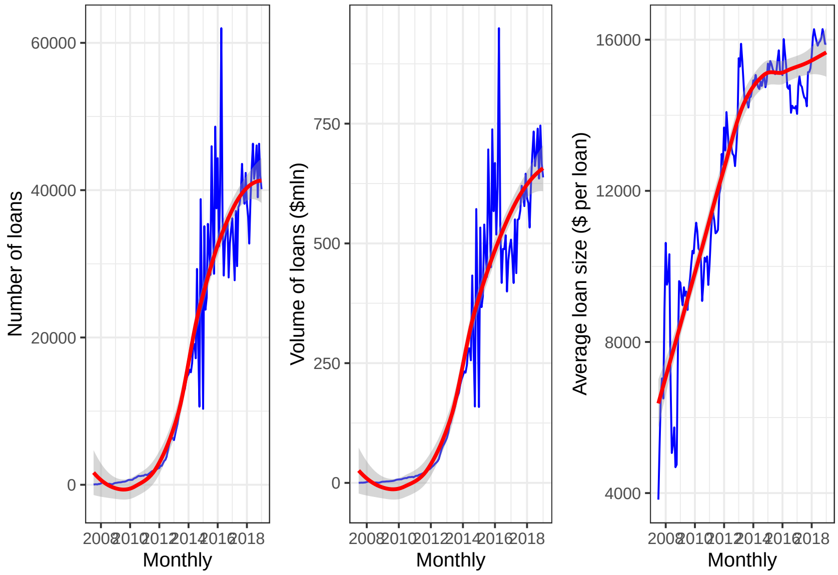
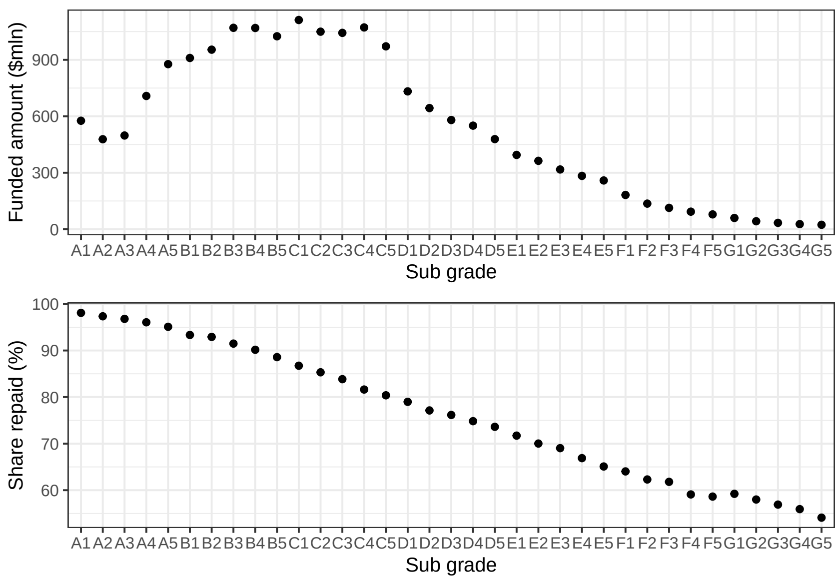


Figure 1.5: Business volume written per month

Table 1.2: Matured loans per status

Loan status	Count	Proportion (%)
Fully Paid	1041952	26048800
Charged Off	261655	6541375
Does not meet the credit policy. Status:Fully Paid	1988	49700
Does not meet the credit policy. Status:Charged Off	761	19025



-1.bb

Figure 1.6: Funding and Write-offs by Sub-grades

(#fig:funded-by-subgrade,)

Appendix

1.5 List of assumptions / limitations regarding the dataset

As mentioned during this report, we had to make numerous assumptions given the lack of clarity of the variable descriptions.

- The dataset does not contain any errors that we cannot notice (e.g. minor error of amount or rate, zipcode).
- The day-1 rating is between A1 and (and no lower than) G5. No note is rated lower than E5 from 6 November 2017, and lower than D5 from 30 June 2019.
- Credit history information for the principal borrower relates to pre-approval and not post-funding. This is clear for the joint applicants, but simply an assumption for the principal borrower.
- *Survival effect*: The dataset does not include applications that were rejected.

We do hope that LendingClub investors receive information of much better quality!

1.6 Data preparation and formatting

We used different sources of information:

- The LendingClub dataset made available on Kaggle;
- US georgraphical data about zip and FIPS codes;
- Market interest rates from the Saint Louis Federal Reserve Bank; and,
- Macro data from the same source.

We here show the code used to prepare the data.

1.6.1 LendinClub dataset

```
local({
  #
  # STEP 1: Download the dataset
  #
  #   Got to https://www.kaggle.com/wendykan/lending-club-loan-data
  #
  #   Download into the 'datasets' subdirectory
  #   Unzip the file.
  #   WARNING: The unzipping will be about 2.4GB
  #
  #   Name the sql database "datasets/lending_club.sqlite"
```

```r
#

#
# STEP 2: Prepare the dabase as a tibble
#

library(RSQLite)
db_conn <-
  dbConnect(RSQLite::SQLite(), "datasets/lending_club.sqlite")
dbListTables(db_conn)

# Returns a 2.96GB data frame
lending_club <- dbGetQuery(db_conn, "SELECT * FROM loan")
lending_club <- as_tibble(lending_club)

# Close the database
dbDisconnect(db_conn)

# Compressed to ca.285MB on disk
saveRDS(lending_club, "datasets/lending_club.rds")


library(tidyverse)
library(lubridate)
library(hablar)

# Before reformat in case the previous step was already done
# lending_club <- readRDS("datasets/lending_club.rds")
#
# str(lending_club)
#

# We leave the original dataset untouched and work with a copy.
lc <- lending_club

lc <- lc %>%
  # Remove useless strings
  mutate(
    term       = str_remove(term, " months"),
    emp_length = str_replace(emp_length, "<1", "0"),
    emp_length = str_replace(emp_length, "10+", "10"),
    emp_length = str_remove(emp_length, "years")
  ) %>%

  # Creates dates out of strings - Parse errors will be raised when no dates.
  mutate(
    debt_settlement_flag_date = as_date(dmy(
      str_c("1-", debt_settlement_flag_date)
    )),
    earliest_cr_line          = as_date(dmy(str_c(
      "1-", earliest_cr_line
    ))),
    hardship_start_date       = as_date(dmy(str_c(
      "1-", hardship_start_date
    ))),
```

```r
    hardship_end_date          = as_date(dmy(str_c(
      "1-", hardship_end_date
    ))),
    issue_d                    = as_date(dmy(str_c("1-", issue_d))),
    last_credit_pull_d         = as_date(dmy(str_c(
      "1-", last_credit_pull_d
    ))),
    last_pymnt_d               = as_date(dmy(str_c(
      "1-", last_pymnt_d
    ))),
    next_pymnt_d               = as_date(dmy(str_c(
      "1-", next_pymnt_d
    ))),
    payment_plan_start_date    = as_date(dmy(str_c(
      "1-", payment_plan_start_date
    ))),
    sec_app_earliest_cr_line   = as_date(dmy(str_c(
      "1-", sec_app_earliest_cr_line
    ))),
    settlement_date            = as_date(dmy(str_c(
      "1-", settlement_date
    )))
  ) %>%

  # Bulk type conversion with convert from the `hablar` package
  convert(
    # Strings
    chr(emp_title, title, url, zip_code),

    # Factors
    fct(
      addr_state,
      application_type,
      debt_settlement_flag,
      desc,
      disbursement_method,
      grade,
      hardship_flag,
      hardship_loan_status,
      hardship_reason,
      hardship_status,
      hardship_type,
      home_ownership,
      id,
      initial_list_status,
      loan_status,
      member_id,
      policy_code,
      purpose,
      pymnt_plan,
      settlement_status,
      sub_grade,
      verification_status,
      verification_status_joint
    ),
```

```
    # Integers
    int(
      acc_now_delinq,
      acc_open_past_24mths,
      chargeoff_within_12_mths,
      collections_12_mths_ex_med,
      deferral_term,
      delinq_2yrs,
      emp_length,
      hardship_dpd,
      hardship_length,
      inq_fi,
      inq_last_12m,
      inq_last_6mths,
      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_last_delinq,
      mths_since_last_major_derog,
      mths_since_last_record,
      mths_since_rcnt_il,
      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,
      open_acc,
      open_acc_6m,
      open_act_il,
      open_il_12m,
      open_il_24m,
      open_rv_12m,
      open_rv_24m,
      sec_app_chargeoff_within_12_mths,
      sec_app_collections_12_mths_ex_med,
      sec_app_inq_last_6mths,
      sec_app_mort_acc,
      sec_app_mths_since_last_major_derog,
      sec_app_num_rev_accts,
```

```
      sec_app_open_acc,
      sec_app_open_act_il,
      term
    ),

    # Floating point
    dbl(
      all_util,
      annual_inc,
      annual_inc_joint,
      avg_cur_bal,
      bc_open_to_buy,
      bc_util,
      collection_recovery_fee,
      delinq_amnt,
      dti,
      dti_joint,
      funded_amnt,
      funded_amnt_inv,
      hardship_amount,
      hardship_last_payment_amount,
      hardship_payoff_balance_amount,
      il_util,
      installment,
      int_rate,
      last_pymnt_amnt,
      loan_amnt,
      max_bal_bc,
      orig_projected_additional_accrued_interest,
      out_prncp,
      out_prncp_inv,
      pct_tl_nvr_dlq,
      percent_bc_gt_75,
      pub_rec,
      pub_rec_bankruptcies,
      recoveries,
      revol_bal,
      revol_bal_joint,
      revol_util,
      sec_app_revol_util,
      settlement_amount,
      settlement_percentage,
      tax_liens,
      tot_coll_amt,
      tot_cur_bal,
      tot_hi_cred_lim,
      total_acc,
      total_bal_ex_mort,
      total_bal_il,
      total_bc_limit,
      total_cu_tl,
      total_il_high_credit_limit,
      total_pymnt,
      total_pymnt_inv,
      total_rec_int,
```

```
        total_rec_late_fee,
        total_rec_prncp,
        total_rev_hi_lim
      )
    ) %>%

    # Converts some values to 1/-1 (instead of Boolean)
    mutate(
      pymnt_plan =           if_else(pymnt_plan == "y",           1, -1),
      hardship_flag =        if_else(hardship_flag == "Y",        1, -1),
      debt_settlement_flag = if_else(debt_settlement_flag == "Y", 1, -1)
    ) %>%

    # Some values are percentages
    mutate(
      int_rate = int_rate / 100,
      dti = dti / 100,
      dti_joint = dti_joint / 100,
      revol_util = revol_util / 100,
      il_util = il_util / 100,
      all_util = all_util / 100,
      bc_open_to_buy = bc_util / 100,
      pct_tl_nvr_dlq = pct_tl_nvr_dlq / 100,
      percent_bc_gt_75 = percent_bc_gt_75 / 100,
      sec_app_revol_util = sec_app_revol_util / 100
    ) %>%

    # Create quasi-centered numerical grades out of grade factors with "A" = 3 down to "G" = -3
    mutate(grade_num = 4 - as.integer(grade)) %>%

    # Ditto with sub_grades. "A1" = +3.4, "A3" = +3.0, down to "G3" = -3.0, "G5" = -3.4
    mutate(sub_grade_num = 3.6 - as.integer(sub_grade) / 5) %>%

    # Keep the first 3 digits of the zipcode as numbers
    mutate(zip_code = as.integer(str_sub(zip_code, 1, 3))) %>%

    # Remove empty columns
    select(-id, -member_id, -url)

  saveRDS(lc, "datasets/lending_club_reformatted.rds")


  # Select loans which have matured or been terminated
  past_loans <- lc %>%
    filter(
      loan_status %in% c(
        "Charged Off",
        "Does not meet the credit policy. Status:Charged Off",
        "Does not meet the credit policy. Status:Fully Paid",
        "Fully Paid"
      )
    )

  saveRDS(past_loans, "datasets/lending_club_reformatted_paid.rds")
})
```

1.6.2 Zip codes and FIPS codes

The R package `zipcode` was installed.

```
#
# ZIPCodes dataset.
#

library(zipcode)
data(zipcode)
zips <- zipcode %>%
  as_tibble() %>%
  mutate(zip = as.integer(str_sub(zip, 1, 3)))

saveRDS(zips, "datasets/zips.rds")
```

A csv file containing zip codes, FIPS codes and population information was downloaded from the *Simple Maps* [8] website.

```
local({
  zipfips <- read.csv("datasets/csv/uszips.csv")

  # Delete the last 2 digits of the zipcode and only keep relevant information
  zipfips <-
    zipfips %>%
    as_tibble() %>%
    mutate(zip = floor(zip / 100)) %>%
    select(zip, state_id, population, county_fips, county_name)

  saveRDS(zipfips, "datasets/zipfips.rds")
})
```

1.6.3 Market interest rates

Market interest rates (3-year and 5-year swap rates) were download from the Saint Louis Federal Reserve Bank. Datasets are split between before and after the LIBOR fixing scandal. The datasets are merged with disctinct dates.

Download sources are:

- Pre-LIBOR 3-y swap https://fred.stlouisfed.org/series/DSWP3
- Post-LIBOR 3-y swap https://fred.stlouisfed.org/series/ICERATES1100USD3Y
- Pre-LIBOR 5-y swap https://fred.stlouisfed.org/series/MSWP5
- Post-LIBOR 5-y swap https://fred.stlouisfed.org/series/ICERATES1100USD5Y

```
local({
  LIBOR3Y <- read.csv("datasets/csv/DSWP3.csv") %>%
    as_tibble() %>%
    filter(DSWP3 != ".") %>%
    mutate(DATE = as_date(DATE),
           RATE3Y = as.numeric(as.character(DSWP3)) / 100) %>%
    select(DATE, RATE3Y)

  ICE3Y <- read.csv("datasets/csv/ICERATES1100USD3Y.csv") %>%
```

[8] https://simplemaps.com/data/us-zips

```r
    as_tibble() %>%
    filter(ICERATES1100USD3Y != ".") %>%
    mutate(DATE = as_date(DATE),
           RATE3Y = as.numeric(as.character(ICERATES1100USD3Y)) / 100) %>%
    select(DATE, RATE3Y)


  LIBOR5Y <- read.csv("datasets/csv/DSWP5.csv") %>%
    as_tibble() %>%
    filter(DSWP5 != ".") %>%
    mutate(DATE = as_date(DATE),
           RATE5Y = as.numeric(as.character(DSWP5)) / 100) %>%
    select(DATE, RATE5Y)

  ICE5Y <- read.csv("datasets/csv/ICERATES1100USD5Y.csv") %>%
    as_tibble() %>%
    filter(ICERATES1100USD5Y != ".") %>%
    mutate(DATE = as_date(DATE),
           RATE5Y = as.numeric(as.character(ICERATES1100USD5Y)) / 100) %>%
    select(DATE, RATE5Y)

  RATES3Y <- LIBOR3Y %>% rbind(ICE3Y) %>%
    arrange(DATE) %>% distinct(DATE, .keep_all = TRUE)

  RATES5Y <- LIBOR5Y %>% rbind(ICE5Y) %>%
    arrange(DATE) %>% distinct(DATE, .keep_all = TRUE)

  saveRDS(RATES3Y, "datasets/rates3Y.rds")
  saveRDS(RATES5Y, "datasets/rates5Y.rds")


  # Note there are 7212 days from 1 Jan 2000 to 30 Sep 2019
  #
  # (ymd("2000-01-01") %--% ymd("2019-09-30")) %/% days(1)
  RATES <- tibble(n = seq(0, 7212)) %>%

    # Create a column with all dates
    mutate(DATE = ymd("2000-01-01") + days(n)) %>%
    select(-n) %>%

    # Add all daily 3- then 5-year rates and fill missing down
    left_join(RATES3Y) %>%
    fill(RATE3Y, .direction = "down") %>%

    left_join(RATES5Y) %>%
    fill(RATE5Y, .direction = "down")

  saveRDS(RATES, "datasets/rates.rds")
})
```

1.6.4 Macro-economical data

Macro-economical datasets were sourced from the same website as Microsoft Excel files. They were converted as-is to tab-separated csv files with LibreOffice.

- Median income per household: https://geofred.stlouisfed.org/map/?th=pubugn&cc=5&rc=false&im=fractile&sb&lng=-112.41&lat=44.31&zm=4&sl&sv&am=Average&at=Not%20Seasonally%20Adjusted,%20Annual,%20Dollars&sti=2022&fq=Annual&rt=county&un=lin&dt=2017-01-01

- Per capita personal income: https://geofred.stlouisfed.org/map/?th=pubugn&cc=5&rc=false&im=fractile&sb&lng=-112.41&lat=44.31&zm=4&sl&sv&am=Average&at=Not%20Seasonally%20Adjusted,%20Annual,%20Dollars&sti=882&fq=Annual&rt=county&un=lin&dt=2017-01-01

- Unemployment: https://geofred.stlouisfed.org/map/?th=rdpu&cc=5&rc=false&im=fractile&sb&lng=-90&lat=40&zm=4&sl&sv&am=Average&at=Not%20Seasonally%20Adjusted,%20Monthly,%20Percent&sti=1224&fq=Monthly&rt=county&un=lin&dt=2019-08-01

```r
local({
  ####################################################################################################
  ##
  ## Median income per household by FIPS from 2002 to 2017
  ##
  medianIncome <-
    # Load the dataset after dropping the first line
    read.csv(
      "datasets/csv/GeoFRED_Estimate_of_Median_Household_Income_by_County_Dollars.csv",
      sep = "\t",
      skip = 1,
      stringsAsFactors = FALSE
    ) %>%

    # Drops columnsn containing a unique identifier and the FIPS name
    select(-"Series.ID", -"Region.Name") %>%

    # Rename the relevant column to 'FIPS'
    rename(FIPS = "Region.Code") %>%

    # Order by FIPS
    arrange(FIPS) %>%

    # Transpose the table to have the FIPS as columns, and the years as observations
    t()

  # After the transpose, correct the name of the columns by using the first row (from which blanks are removed)
  colnames(medianIncome) <- medianIncome[1, ] %>% str_remove("[ ]")

  # Create a new column with the name of the year and converts to tible
  medianIncome <-
    cbind(Year = rownames(medianIncome), medianIncome) %>%
    as_tibble() %>%
    slice(2:n())

  # Converts the content of the Year column to an actual date
  medianIncome <- medianIncome %>%
    mutate(Year = str_replace(Year, "[X]", ""),
           Year = ymd(str_c(Year, "-12-31")))

  saveRDS(medianIncome, "datasets/medianincome.rds")


  ####################################################################################################
```

```r
##
## Per capita income by FIPS from 2002 to 2017
##
personalIncome <-
  # Load the dataset after dropping the first line
  read.csv(
    "datasets/csv/GeoFRED_Per_Capita_Personal_Income_by_County_Dollars.csv",
    sep = "\t",
    skip = 1,
    stringsAsFactors = FALSE
  ) %>%

  # Drops columnsn containing a unique identifier and the FIPS name
  select(-"Series.ID", -"Region.Name") %>%

  # Rename the relevant column to 'FIPS'
  rename(FIPS = "Region.Code") %>%

  # Order by FIPS
  arrange(FIPS) %>%

  # Transpose the table to have the FIPS as columns, and the years as observations
  t()

# Correct the name of the columns after the transpose
colnames(personalIncome) <-
  personalIncome[1, ] %>% str_remove("[ ]")

# Create a new column with the name of the year and converts to tible
personalIncome <-
  cbind(Year = rownames(personalIncome), personalIncome) %>%
  as_tibble() %>%
  slice(2:n())

# Converts the content of the Year column to an actual date
personalIncome <- personalIncome %>%
  mutate(Year = str_replace(Year, "[X]", ""),
         Year = ymd(str_c(Year, "-12-31")))

saveRDS(personalIncome, "datasets/personalincome.rds")


###################################################################################################
##
## Unemplyment rate monthly by FIPS from January 2000 to August 2019
##
unemploymentRate <-
  # Load the dataset after dropping the first line
  read.csv(
    "datasets/csv/GeoFRED_Unemployment_Rate_by_County_Percent.csv",
    sep = "\t",
    skip = 1,
    stringsAsFactors = FALSE
  ) %>%

```

```r
    # Drops columnsn containing a unique identifier and the FIPS name
    select(-"Series.ID", -"Region.Name") %>%

    # Rename the relevant column to 'FIPS'
    rename(FIPS = "Region.Code") %>%

    # Order by FIPS
    arrange(FIPS) %>%

    # Transpose the table to have the FIPS as columns, and the years as observations
    t()

  # Correct the name of the columns after the transpose
  colnames(unemploymentRate) <-
    unemploymentRate[1, ] %>% str_remove("[ ]")

  # Create a new column with the name of the year and converts to tible
  unemploymentRate <-
    cbind(Year = rownames(unemploymentRate), unemploymentRate) %>%
    as_tibble() %>%
    slice(2:n())

  # Converts the content of the Year column to an actual date
  unemploymentRate <- unemploymentRate %>%
    mutate(
      Year = str_replace(Year, "[X]", ""),
      Year = str_replace(Year, "[.]", "-"),
      Year = ymd(str_c(Year, "-1"))
    )

  saveRDS(unemploymentRate, "datasets/unemployment.rds")
})
```

1.7 List of variables

This table presents the list of variables provided in the original dataset. The descriptions come from a spreadsheet attached with the dataset and, unfortunately, are not extremely precise and subject to interpretation. We added comments and/or particular interpretations in *CAPITAL LETTERS*.

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle

Variable Name	Description
loan_amnt	The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.
funded_amnt	The total amount committed to that loan at that point in time.
funded_amnt_inv	The total amount committed by investors for that loan at that point in time.
term	The number of payments on the loan. Values are in months and can be either 36 or 60.
int_rate	Interest Rate on the loan

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle *(continued)*

Variable Name	Description
installment	The monthly payment owed by the borrower if the loan originates.
grade	LC assigned loan grade
sub_grade	LC assigned loan subgrade
emp_title	The job title supplied by the Borrower when applying for the loan.
emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
home_ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER
annual_inc	The self-reported annual income provided by the borrower during registration.
verification_status	Indicates if income was verified by LC, not verified, or if the income source was verified
issue_d	The month which the loan was funded
loan_status	Current status of the loan
pymnt_plan	Indicates if a payment plan has been put in place for the loan
url	URL for the LC page with listing data.
desc	Loan description provided by the borrower
purpose	A category provided by the borrower for the loan request.
title	The loan title provided by the borrower
zip_code	The first 3 numbers of the zip code provided by the borrower in the loan application.
addr_state	The state provided by the borrower in the loan application
dti	A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income.
delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
earliest_cr_line	The month the borrower's earliest reported credit line was opened
inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
mths_since_last_delinq	The number of months since the borrower's last delinquency.
mths_since_last_record	The number of months since the last public record.
open_acc	The number of open credit lines in the borrower's credit file.
pub_rec	Number of derogatory public records
revol_bal	Total credit revolving balance

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle *(continued)*

Variable Name	Description
revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
total_acc	The total number of credit lines currently in the borrower's credit file
initial_list_status	The initial listing status of the loan. Possible values are – W, F
out_prncp	Remaining outstanding principal for total amount funded
out_prncp_inv	Remaining outstanding principal for portion of total amount funded by investors
total_pymnt	Payments received to date for total amount funded
total_pymnt_inv	Payments received to date for portion of total amount funded by investors
total_rec_prncp	Principal received to date
total_rec_int	Interest received to date
total_rec_late_fee	Late fees received to date
recoveries	Post charge off gross recovery
collection_recovery_fee	Post charge off collection fee
last_pymnt_d	Last month payment was received
last_pymnt_amnt	Last total payment amount received
next_pymnt_d	Next scheduled payment date
last_credit_pull_d	The most recent month LC pulled credit for this loan
collections_12_mths_ex_med	Number of collections in 12 months excluding medical collections
mths_since_last_major_derog	Months since most recent 90 - day or worse rating
policy_code	Publicly available policy_code = 1 / New products not publicly available policy_code = 2
application_type	Indicates whether the loan is an individual application or a joint application with two co - borrowers
annual_inc_joint	The combined self - reported annual income provided by the co - borrowers during registration
dti_joint	A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income
verification_status_joint	Indicates if income was verified by LC, not verified, or if the income source was verified
acc_now_delinq	The number of accounts on which the borrower is now delinquent.
tot_coll_amt	Total collection amounts ever owed
tot_cur_bal	Total current balance of all accounts
open_acc_6m	Number of open trades in last 6 months
open_act_il	Number of currently active installment trades
open_il_12m	Number of installment accounts opened in past 12 months
open_il_24m	Number of installment accounts opened in past 24 months

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle *(continued)*

Variable Name	Description
mths_since_rcnt_il	Months since most recent installment accounts opened
total_bal_il	Total current balance of all installment accounts
il_util	Ratio of total current balance to high credit / credit limit on all install acct
open_rv_12m	Number of revolving trades opened in past 12 months
open_rv_24m	Number of revolving trades opened in past 24 months
max_bal_bc	Maximum current balance owed on all revolving accounts
all_util	Balance to credit limit on all trades
total_rev_hi_lim	Total revolving high credit / credit limit
inq_fi	Number of personal finance inquiries
total_cu_tl	Number of finance trades
inq_last_12m	Number of credit inquiries in past 12 months
acc_open_past_24mths	Number of trades opened in past 24 months.
avg_cur_bal	Average current balance of all accounts
bc_open_to_buy	Total open to buy on revolving bankcards.
bc_util	Ratio of total current balance to high credit / credit limit for all bankcard accounts.
chargeoff_within_12_mths	Number of charge - offs within 12 months
delinq_amnt	The past - due amount owed for the accounts on which the borrower is now delinquent.
mo_sin_old_il_acct	Months since oldest bank installment account opened
mo_sin_old_rev_tl_op	Months since oldest revolving account opened
mo_sin_rcnt_rev_tl_op	Months since most recent revolving account opened
mo_sin_rcnt_tl	Months since most recent account opened
mort_acc	Number of mortgage accounts.
mths_since_recent_bc	Months since most recent bankcard account opened.
mths_since_recent_bc_dlq	Months since most recent bankcard delinquency
mths_since_recent_inq	Months since most recent inquiry.
mths_since_recent_revol_delinq	Months since most recent revolving delinquency.
num_accts_ever_120_pd	Number of accounts ever 120 or more days past due
num_actv_bc_tl	Number of currently active bankcard accounts
num_actv_rev_tl	Number of currently active revolving trades
num_bc_sats	Number of satisfactory bankcard accounts
num_bc_tl	Number of bankcard accounts
num_il_tl	Number of installment accounts
num_op_rev_tl	Number of open revolving accounts
num_rev_accts	Number of revolving accounts
num_rev_tl_bal_gt_0	Number of revolving trades with balance > 0
num_sats	Number of satisfactory accounts
num_tl_120dpd_2m	Number of accounts currently 120 days past due (updated in past 2 months)
num_tl_30dpd	Number of accounts currently 30 days past due (updated in past 2 months)
num_tl_90g_dpd_24m	Number of accounts 90 or more days past due in last 24 months

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle *(continued)*

Variable Name	Description
num_tl_op_past_12m	Number of accounts opened in past 12 months
pct_tl_nvr_dlq	Percent of trades never delinquent
percent_bc_gt_75	Percentage of all bankcard accounts > 75 % of limit.
pub_rec_bankruptcies	Number of public record bankruptcies
tax_liens	Number of tax liens
tot_hi_cred_lim	Total high credit/credit limit
total_bal_ex_mort	Total credit balance excluding mortgage
total_bc_limit	Total bankcard high credit/credit limit
total_il_high_credit_limit	Total installment high credit/credit limit
revol_bal_joint	Total credit revolving balance
sec_app_earliest_cr_line	Earliest credit line at time of application for the secondary applicant
sec_app_inq_last_6mths	Credit inquiries in the last 6 months at time of application for the secondary applicant
sec_app_mort_acc	Number of mortgage accounts at time of application for the secondary applicant
sec_app_open_acc	Number of open trades at time of application for the secondary applicant
sec_app_revol_util	Ratio of total current balance to high credit/credit limit for all revolving accounts
sec_app_open_act_il	Number of currently active installment trades at time of application for the secondary applicant
sec_app_num_rev_accts	Number of revolving accounts at time of application for the secondary applicant
sec_app_chargeoff_within_12_mths	Number of charge-offs within last 12 months at time of application for the secondary applicant
sec_app_collections_12_mths_ex_med	Number of collections within last 12 months excluding medical collections at time of application for the secondary applicant
sec_app_mths_since_last_major_derog	Months since most recent 90-day or worse rating at time of application for the secondary applicant
hardship_flag	Flags whether or not the borrower is on a hardship plan
hardship_type	Describes the hardship plan offering
hardship_reason	Describes the reason the hardship plan was offered
hardship_status	Describes if the hardship plan is active, pending, canceled, completed, or broken
deferral_term	Amount of months that the borrower is expected to pay less than the contractual monthly payment amount due to a hardship plan
hardship_amount	The interest payment that the borrower has committed to make each month while they are on a hardship plan
hardship_start_date	The start date of the hardship plan period
hardship_end_date	The end date of the hardship plan period

Table 1.3: Description of the dataset variables as provided in the dataset downloaded from Kaggle *(continued)*

Variable Name	Description
payment_plan_start_date	The day the first hardship plan payment is due. For example, if a borrower has a hardship plan period of 3 months, the start date is the start of the three-month period in which the borrower is allowed to make interest-only payments.
hardship_length	The number of months the borrower will make smaller payments than normally obligated due to a hardship plan
hardship_dpd	Account days past due as of the hardship plan start date
hardship_loan_status	Loan Status as of the hardship plan start date
orig_projected_additional_accrued_interest	The original projected additional interest amount that will accrue for the given hardship payment plan as of the Hardship Start Date. This field will be null if the borrower has broken their hardship payment plan.
hardship_payoff_balance_amount	The payoff balance amount as of the hardship plan start date
hardship_last_payment_amount	The last payment amount as of the hardship plan start date
disbursement_method	The method by which the borrower receives their loan. Possible values are: CASH, DIRECT_PAY
debt_settlement_flag	Flags whether or not the borrower, who has charged-off, is working with a debt-settlement company.
debt_settlement_flag_date	The most recent date that the Debt_Settlement_Flag has been set
settlement_status	The status of the borrower’s settlement plan. Possible values are: COMPLETE, ACTIVE, BROKEN, CANCELLED, DENIED, DRAFT
settlement_date	The date that the borrower agrees to the settlement plan
settlement_amount	The loan amount that the borrower has agreed to settle for
settlement_percentage	The settlement amount as a percentage of the payoff balance amount on the loan

1.8 System version

```
##                                        sysname
##                                        "Linux"
##                                        release
##                             "5.3.0-20-generic"
##                                        version
## "#21-Ubuntu SMP Wed Oct 23 16:20:37 UTC 2019"
##                                       nodename
##                                         "x260"
##                                        machine
##                                       "x86_64"
##                                          login
##                                      "unknown"
```

```
##                                           user
##                                     "emmanuel"
##                                 effective_user
##                                     "emmanuel"
```

Bibliography

Kim, A. and Cho, S.-B. (2019). An ensemble semi-supervised learning method for predicting defaults in social lending. *Engineering Applications of Artificial Intelligence*, 81:193–199.

LendingClub Corporation (San Francisco, C. (2019). Prospectus regulatory filing s3-asr for member payment dependent notes. https://www.sec.gov/Archives/edgar/data/1409970/000140997019000988/0001409970-19-000988-index.htm. [Note: Accessed 31 October 2019].

Peng, R. (2012). *Exploratory data analysis with R*. Lulu. com.