

Working Paper № 2

In this stage, we focused on understanding and cleaning the data. Our main focus was on the final outcome of each loan. We noticed that all loans started in 2016, and the last data snapshot is from 2019. Consequently, we started by removing (filtering) loans that exceeded 36 months (3 years) (otherwise we don't have the loan's final outcome). Secondly, we removed all loans that their status was different from "Fully Paid" or "Charged Off" in the 2019 snapshot. These two steps reduced the number of loans from 434,407 to 269,583 and this is the "new data" that will be used for the EDA. Thirdly, we realized that there are some leakage features – features that their values change through the course of the loan. Accordingly, we created 3 main functions that helped us further investigate the data (see the attached Rmd and Python files):

- **Comparing 2018-2019 tables** (see function [match_cols¹](#) in R rmd file / [matching_columns²](#) in Python ipynb file): this function compares each column in 2018 with its parallel column in 2019. The goal of this function is to identify the leakage variables:
 - Leakage / "varying" columns: columns that their values are updated throughout the loan can't be used in our prediction model. The reason: on the day the loans were issued, there's a high chance they had a different value / no value at all.
We found and dealt with 52 "varying" columns (see **Table 3: Matching columns** in the appendix).
 - Consistent columns: columns that their value is or should be consistent through the course of a loan. Some columns that should have been consistent had a tiny inconsistency, for example, the "interest rate" (83 cases). Thus, we built a python function that presents those unmatched rows - so we can inspect them closely and decide the proper way to deal with them (see function ["unmatched_rows_one_table"](#) in the python file).
- **NAs counter** (see function [NAs_count](#) in R rmd file and [see_rows_with_NaN](#) in Python) – this R function counts how many NAs each column has, helping us detect columns that might be redundant in case most of the loans in that column have NA values. Otherwise, if the column has little amount of NAs we would like to find the best way to deal with them. For this purpose we built a python function ["see_rows_with_NaN"](#) - the input is a certain column and the output is all the rows with NaN values. This function enables us to explore if the other values in the other different columns also hold NaN values and thus encouraging us to filter those rows.
- **Cleaning function** (see function [clean_data](#) in R): gets as an input a table and performs all the cleaning actions we will present in this Working Paper and the appendix (table 1 and especially table 2). Moreover, it creates the **realized return column**. The realized return will be calculated according to the following equation:

$$Realized\ Return = \frac{p-f}{f} \times \frac{12}{t}$$

p = full amount recovered from the loan = total_pymnt_inv (total payment that the investor got)

¹ [R file](#)

² [Python file](#)

f = full amount invested in a loan = funded_amnt_inv

t = Nominal duration of the loan in months = column term (=36 month)

We will present 3 main steps we used in this stage for cleaning the data:

1. **Exploratory Data Analysis (EDA):**

To understand and clean the data, we explored the columns from different perspectives:

- A. NAs Count and analysis: Columns that most of their values are NAs are good candidates for filtering. The threshold we used for filtering is 2% - if more than 2% of a column's value are NAs - this column will be a good candidate for filtering. Else, if the column is too important to remove, we explored the distribution of the feature and decided the right way to deal with those NAs (see part 3 of **Altering + Cleaning columns**)
- B. Exploration of the distribution of the features + Scatter plots of a feature and the target column (realized return) and identify occurrences of outliers. We also used functions as `tables` and `summary` in **R** to explore each feature.
- C. Density plot of the target variable (Realized return) given the different classes of a discrete variable.
- D. Correlation Analysis: We may want to filter columns with low correlation with our target variable or features that have high correlation with each other.³ We believe that feature selection is a more appropriate tool for filtering columns in the claim they don't have predicting power. (We wish to address this in the following stage [building a model]).

* The full EDA is presented in the rmd file.

The EDA step has provided us a better understanding of the data and helped us in the following two steps:

2. **Removing columns**: The given data has 150 different features, many of those are irrelevant or shouldn't be used for several different reasons.

We Categorized **5** main reasons for filtering columns:

- A. Business-wise: The column is irrelevant from the business perspective.
- B. The column is redundant: There is a similar/identical column or there is a high correlation with another column.
- C. Too many NA's / Missing Values⁴.
- D. "Varying Column": There are several columns that hold "updating" information, meaning that on the day the loan was issued (In 2016), the column held a different value. That means we can't use that column in our model (as we don't know the values of the loans for this column on the day the loan was issued)⁵.

³ We may take an action only if the correlation is very High ($>\sim 0.99$) or very small ($<\sim 0.01$), due to the fact that a feature might have little correlation with the target variable. However, a combination of some features might have good predicting power.

⁴ For example, all the columns that are related to joint application, settlement plan and hardship plan - Most of the loans had NAs.

⁵ In order to know which columns are varying, we have built a function in R and Python (see function `match_cols` / `matching_columns` in rmd and ipynb files respectively). The result of the function is presented in table 3 in column number 3 "Matching Rows (between 2018 to 2019)" in the appendix.

E. Other reasons.

We went over all of the features and ended up removing **92** different features.

See Table 1: *Features to remove* in the appendix for the full table of the columns we filtered and the reasons for filtering each and every one of them.

Important notes:

- We are certain that we will remove more features in the following stage of building a model - we will use “feature selection” and may find features that don’t have predicting power towards the target variable.
 - We might retrieve columns that were filtered in this step, for instance, if we find them contributing to the predicting model.
3. **Altering + Cleaning columns:** Regarding the **59** columns that were left (after filtering columns in stage 2), some of those required cleaning. We conducted **4** different types of cleaning on those columns:
- A. Dealing with NA’s / Missing Values: For columns with NAs, we had to determine the method to deal with them. We can either filter those rows (on the price of losing the whole loan’s data) or change the NA values (replace the NA with mean/**mode** values for example).
 - B. Altering the column’s type: Some columns were wrongly parsed, therefore we had to alter those column types.⁶
 - C. Altering specific values: We may want to replace specific values in various columns.⁷
 - D. Dealing with Outliers. We might consider removing rows with outlying values or alter those values.⁸

See Table 2: *Cleaning Methods* in the appendix for the full table of the cleaning methods we used on each and every column that is not mentioned in table 1 (all the features we haven’t filtered).

- Some columns were necessary for calculating the realized return and thus were not filtered in stage 1 (table 1) even though we won’t use them in the model.
- In the following stages we may locate more “problems” within the columns that were not filtered (the columns mentioned in table 2 in the appendix) and may need to conduct more cleaning.

⁶ For example, from string to number or from string to date.

⁷ For example, changing column “term” values: from “36 months” to 36 and as integer.

⁸ For example, in the “purpose” column, one value was “wedding”, we treated this value as an outlier and changed its value to “other”.

Appendixes

Table 1: Features to remove

Removing Features Legend:

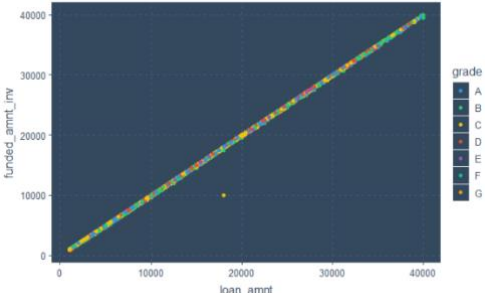
There are 5 main reasons for removing features:

1. **Business-wise:** The column is irrelevant from the business perspective - As the analytical team we can tell for some specific columns that they won't be useful for our prediction.
2. **The column is redundant:** There is a similar/identical column or there is a high correlation with another column.
3. **Too many NAs / Missing Values.**
4. **Leakage variables / "Varying Column"** = There are several columns that hold "updating" information, meaning that on the day the loan was issued (In 2016), the column held a different value. That means we can't use that column in our model (as we don't know the values of the loans for this column in the day the loan was issued).
5. **Other reason**

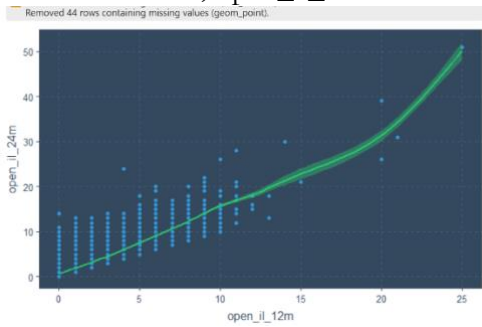
** In the following table - the number in the "reason for removing" column indicates the number of reasons as mentioned above.

** col numbers that are colored in pink are columns that we decided to filter in this stage, even though we believe we might want to consider re-entering them in the model / check if they can increase the model's predicting power.

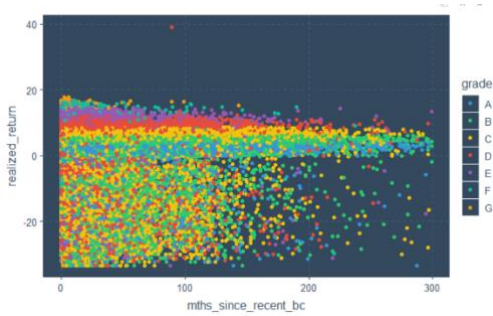
| Column Number | Column Name | Reason for removing (See <i>Removing Feature Legend</i>) | Notes |
|---------------|-----------------|--|--|
| 2 | member_id | 1 + 3 | Empty column + Not relevant from Business perspective |
| 4 | funded_amnt | 2 | Column C (loan_amnt) has identical values (except for 3 rows in Q2). |
| 5 | funded_amnt_inv | 2 | Linear dependent with loan_amnt Scatter plot of funded_amnt VS loan_amnt: |

| | | | |
|----|------------------------|-------|---|
| | | |  <p>Moreover, regarding the day the loan was issued - this column had the value 0 - as nothing was given to the borrower. The loan has just been issued and thus no one gave him money yet. Thus we can't use this column in the predicting model.</p> <p>**This column helped us calculate the realized return.</p> |
| 11 | emp_title | 3 | 19,525 NAs (7.24% of the data) + Too many distinct values - more than 100k unique levels. |
| 12 | emp_length | 1 + 3 | 19,395 NA's (7.19% of the data) |
| 15 | verification_status | 4 | There are 7021 unmatching loans' verification status between 2018 and 2019. This is an indication that the verification_status may change throughout the course of a loan, meaning we can't assume the value we have in this column in 2019 was the same on the day the loan was given. |
| 18 | pymnt_plan | 2 | After filtering - for loans that are Fully Paid / Charged Off the value of the column is "n". Meaning the column is redundant. |
| 19 | url | 1 | Not relevant from the business perspective. |
| 20 | desc | 1 + 3 | 269,573 NA's |
| 22 | title | 2 + 3 | Hold similar information as column "purpose" and has 14,615 NA's |
| 23 | zip_code | 1 + 2 | Hold similar information as column "addr_state" and can't be analyzed from business perspective |
| 28 | fico_range_low | 2 | The difference between fico_range_low to fico_range_high is quite similar for all of the loans, and is up to 5 points. That means that this column holds similar information as column fico_range_high and thus redundant. Moreover the significance of the difference between the thigh and low fico score is negligible since the borrowers still have the same fico status. >580 = poor ; 580-669 = fair ; 670-739 = good ; 740 - 799 = very good ; 800 = exceptional |
| 31 | mths_since_last_delinq | 3 | 127,175 NA's |
| 32 | mths_since_last_record | 3 | 216,551 NA's |

| | | | |
|----|------------------------------|-------|---|
| 39 | out_prncp | 1 | The feature is not necessary for the future model. However, we will probably need it for expected return calculations. |
| 40 | out_prncp_inv | 5 | We do not think that the column is necessary, since it is something that changes during loan lifespan. |
| 41 | total_pymnt | 2 | Redundant to column 42 |
| 43 | total_rec_prncp | 2 | Sum of the columns 43,44,45, and 46 gives us the value of the column 42. |
| 44 | total_rec_int | 2 | Sum of the columns 43,44,45, and 46 gives us the value of the column 42. |
| 45 | total_rec_late_fee | 2 | Sum of the columns 43,44,45, and 46 gives us the value of the column 42. |
| 46 | recoveries | 2 | Sum of the columns 43,44,45, and 46 gives us the value of the column 42. |
| 47 | collection_recovery_fee | 1 | |
| 49 | last_pymnt_amnt | 1 | |
| 50 | next_pymnt_d | 3 | 269,583 NA's |
| 51 | last_credit_pull_d | 1 | |
| 52 | last_fico_range_high | 4 | |
| 53 | last_fico_range_low | 2+3+4 | The borrowers with the lowest FICO (499) in column 52 get 0 in this column. Up to 5 point difference between 53 and 52. |
| 55 | mths_since_last_maj_or_derog | 3 | 190,451 NA's |
| 56 | policy_code | 1 | There is only one value for this column: "1". |
| 58 | annual_inc_joint | 3 | 265,442 NA's 98.46% of the data are individual loans. Therefore we are going to remove all the columns which are related to a joint type of loan (4,141 loans of this kind). |
| 59 | dti_joint | 3 | 265,442 NA's 98.46% of the data are individual loans. Therefore we are going to remove all the columns which are related to a joint type of loan (4,141 loans of this kind). |
| 60 | verification_status_joint | 1 + 3 | 265,442 NA's 98.46% of the data are individual loans. Therefore we are going to remove all the columns which are related to a joint type of loan (4,141 loans of this kind). |
| 62 | tot_coll_amt | 1 | |

| | | | |
|-----------------|----------------------|---|--|
| 65 ⁹ | open_act_il | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 66 | open_il_12m | 2 | Scatter plot of open_il_24m VS open_il_12m: We can see there is a clear correlation between the two variables, as the open_il_12m increases so do the open_il_24m. Therefore we need to filter one of them! We decided for now to filter the column; open_il_12m  |
| 68 | mths_since_rcnt_il | 3 | 8,149 NA's |
| 70 | il_util | 3 | 37,559 NA's |
| 71 | open_rv_12m | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 72 | open_rv_24m | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 74 | all_util | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 77 | total_cu_tl | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 79 | acc_open_past_24mths | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 80 | avg_cur_bal | 1 | We will use this feature only if we will need a number of accounts for a future model. In this stage, we do not find the column useful. |
| 81 | bc_open_to_buy | 3 | 3,104 NA's For now we prefer to filter the column, however we may want to consider reentering the column at the cost of filtering 3,104 loans with NA value in this column. |

⁹ Column numbers that are coloured in pink are columns that we decided to filter in this stage, even though we believe we might want to consider re-entering them in the model / check if they can increase the model's predicting power.

| | | | |
|-----|--------------------------------|---|--|
| 82 | bc_util | 3 | 3,240 NA's For now we prefer to filter the column, however we may want to consider reentering the column at the cost of filtering 3,240 loans with NA value in this column. |
| 85 | mo_sin_old_il_acct | 3 | 8,110 NA's |
| 90 | mths_since_recent_bc | 3 | 2,921 NA's For now we prefer to filter the column, however we may want to consider reentering the column at the cost of filtering 2,921 loans with NA value in this column.  |
| 91 | mths_since_recent_bc_dlq | 3 | 200,380 NA's |
| 92 | mths_since_recent_inq | 3 | 27,778 NA's |
| 93 | mths_since_recent_revol_delinq | 3 | 171,477 NA's |
| 96 | num_actv_rev_tl | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 102 | num_rev_tl_bal_gt_0 | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 104 | num_tl_120dpd_2m | 3 | 12,330 NA's. |
| 105 | num_tl_30dpd | 2 | Column 104 includes this column. |
| 108 | pct_tl_nvr_dlq | 5 | After a deep search, we do not fully understand the meaning of “trade/s” in this context. Due to this issue we are not going to use this column, at least for now. |
| 109 | percent_bc_gt_75 | 3 | 3,127 NAs For now we prefer to filter the column, however we may want to consider reentering the column at the cost of filtering 4,665 loans with NA value in this column. |
| 116 | revol_bal_joint | 3 | 269,583 NA's (100%) |
| 117 | sec_app_fico_range_low | 3 | 269,583 NA's (100%) |

| | | | |
|------------|-------------------------------------|-------|--|
| 118 | sec_app_fico_range_high | 3 | 269,583 NA's (100%) |
| 119 | sec_app_earliest_cr_line | 3 | 269,583 NA's (100%) |
| 120 | sec_app_inq_last_6mths | 3 | 269,583 NA's (100%) |
| 121 | sec_app_mort_acc | 3 | 269,583 NA's (100%) |
| 122 | sec_app_open_acc | 3 + 5 | 269,583 NA's (100%) After a deep search, we do not fully understand the meaning of "trade/s" in this context. |
| 123 | sec_app_revol_util | 3 | 269,583 NA's (100%) After a deep search, we do not fully understand the meaning of "trade/s" in this context. |
| 124 | sec_app_open_act_il | 3 + 5 | 269,583 NA's (100%) |
| 125 | sec_app_num_rev_accounts | 3 | 269,583 NA's (100%) |
| 126 | sec_app_chargeoff_within_12_mths | 3 | 269,583 NA's (100%) |
| 127 | sec_app_collections_12_mths_ex_med | 3 | 269,583 NA's (100%) |
| 128 | sec_app_mths_since_last_major_derog | 3 | 269,583 NA's (100%) |
| 129 | hardship_flag | 3 | 2,168 (~0.8%) of all borrowers have ever been on a hardship plan. Therefore we are going to remove all the columns which are related to a hardship plan. |
| 130 | hardship_type | 3 | 267,415 NA's (99.19%). |
| 131 | hardship_reason | 3 | 267,415 NA's (99.19%). |
| 132 | hardship_status | 3 | 267,415 NA's (99.19%). |
| 133 | deferral_term | 3 | 267,415 NA's (99.19%). |
| 134 | hardship_amount | 3 | 267,415 NA's (99.19%). |
| 135 | hardship_start_date | 3 | 267,415 NA's (99.19%). |
| 136 | hardship_end_date | 3 | 267,415 NA's (99.19%). |
| 137 | payment_plan_start_date | 3 | 267,415 NA's (99.19%). |
| 138 | hardship_length | 3 | 267,415 NA's (99.19%). |

| | | | |
|-----|--|---|---|
| 139 | hardship_dpd | 3 | 267,415 NA's (99.19%). |
| 140 | hardship_loan_status | 3 | 267,415 NA's (99.19%). |
| 141 | orig_projected_additional_accrued_interest | 3 | 267,415 NA's (99.19%). |
| 142 | hardship_payoff_balance_amount | 3 | 267,415 NA's (99.19%). |
| 143 | hardship_last_payment_amount | 3 | 267,415 NA's (99.19%). |
| 144 | disbursement_method | 1 | The method by which the borrowers receive their loan is not significant. |
| 145 | debt_settlement_flag | 5 | Only 8,552 (3.17%) of all borrowers have been working with a debt-settlement company. Therefore we are going to remove all the columns which are related to a settlement. |
| 146 | debt_settlement_flag_date | 3 | 261,031 NA's (96.83%). |
| 147 | settlement_status | 3 | 261,031 NA's (96.83%). |
| 148 | settlement_date | 3 | 261,031 NA's (96.83%). |
| 149 | settlement_amount | 3 | 261,031 NA's (96.83%). |
| 150 | settlement_percentage | 3 | 261,031 NA's (96.83%). |
| 151 | settlement_term | 3 | 261,031 NA's (96.83%). |

Some additional notes regarding the columns that are associated with hardship plan/debt settlement/joint application:

- Joint application - only 2.02% of the loans (8,790 out of 434,407) are of joint type. Therefore all the columns that relate to joint application (57-59; 115-127) are redundant (they are full with NAs).
- Hardship plan - only 1.06% of the loans (4,604 out of 434,407) have been on a hardship plan. Therefore most of the loans have "NA" value in those columns, thus all the columns that are associated with the hardship plan (128-142) will be deleted due to a high number of NAs.
- Debt settlement - only 2.89% of the loans (12,574 out of 434,407) had a settlement plan. Therefore all the columns that are associated with the debt settlement (144-150) will be deleted due to a high number of NAs.

Table 2: Cleaning Methods

In this table we will present all the columns that we will **be using** in our model or in our calculations (such as when calculating the realized return).

Cleaning Features Legend:

There are 6 main Cleaning options that we will use:

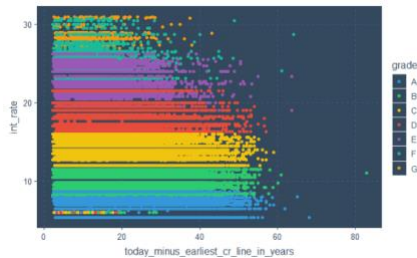
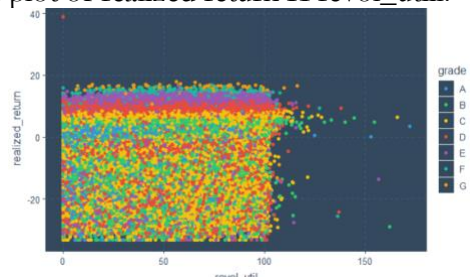
1. **Dealing with NA's / Missing Values:** For columns with NA's we had to determine the method to deal with the missing values. We can either filter those rows (on the price of losing the whole loan's data) or change the NA values (replace the NA with mean | mode values for example)
2. **Altering the column's type:** Some columns were wrongly parsed, therefore we had to Alter those columns types (for example from string to date).
3. **Altering specific values:** We may want to replace specific values in various columns.
4. **Dealing with Outliers:** We might consider removing rows with outlying values or alter the values.
5. **Filtering values**
6. **Using the column to build a new column** (Feature Engineering)

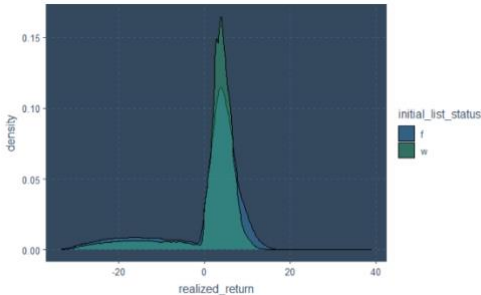
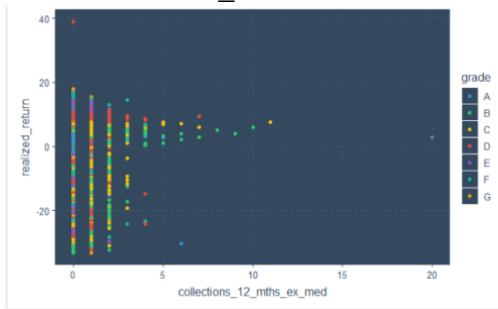
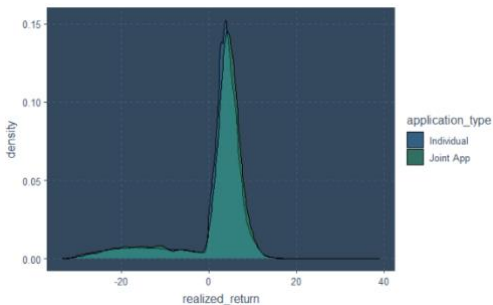
** In the following table - the number in the "Cleaning Method" column indicates the number of reasons as mentioned above.

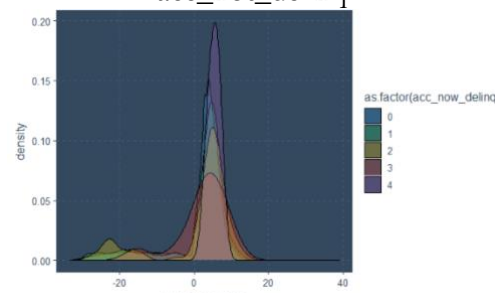
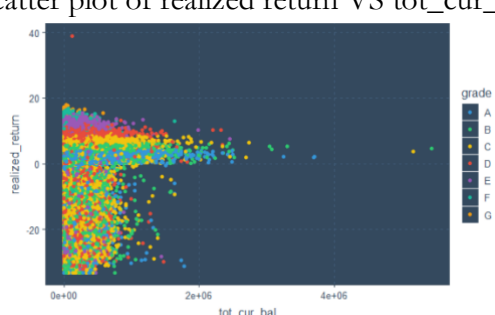
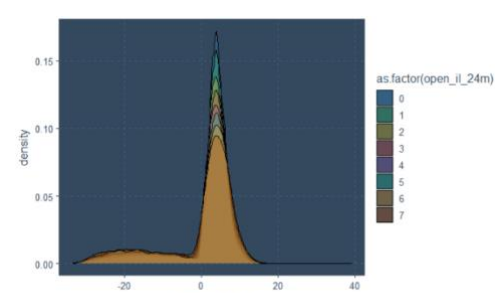
** "Clean Free" = No cleaning was done in this stage regarding the given column.



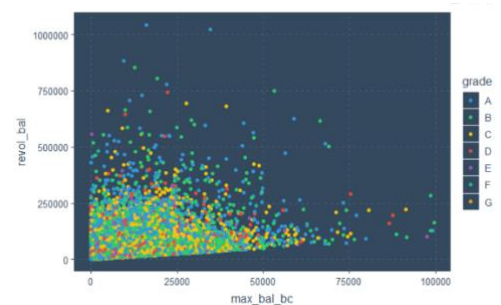
| Column Number | Column Name | Cleaning Method (See <i>Cleaning Feature Legend</i>) | Notes |
|---------------|-------------|--|--|
| 1 | id | Clean Free | No need to clean in any way this is Primary Key. |
| 3 | loan_amnt | Clean Free | |
| 6 | term | 2+3 5 | Alter "36 months" to 36 and "60 months" to 60 and change the column type to integer This feature is important only for filtering the loans that are for 60 months. After Future filtering, this feature we be redundant (As all the loans will be of 36 months) |
| 7 | int_rate | 2+3 | Remove the % sign and change the column type to float. Filter all loans with int_rate = 6. We noticed that there are loans, from all grades, that have int_rate =6. This can't be the true int_rate. |


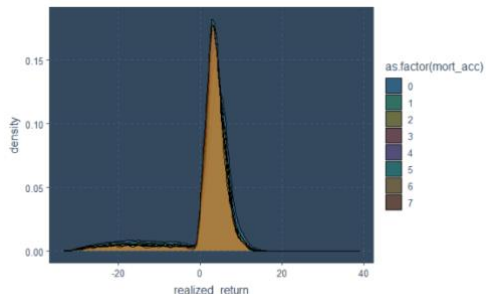
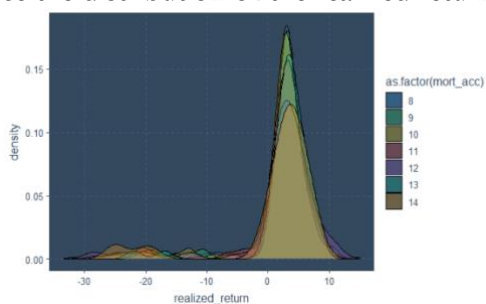
| | | | Moreover many of those rows had different value in the corresponding column in the table of 2018, Therefore we decided to filter them. | | | | | | | | | | | | |
|--------------|------------------|----------------------|---|--------------|---------|--------|-----------|---------|--------|-----|-------|-------|-------|-------|---------|
| 8 | installment | Clean Free | | | | | | | | | | | | | |
| 9 | grade | Clean Free | This feature is important for building the “base model”, we may not use this feature for our “fancier” model as we believe that the grade column is an outcome of the company’s “home model” which already used other given columns. | | | | | | | | | | | | |
| 10 | sub_grade | Clean Free | This feature is important for building the “base model”, we may not use this feature for our “fancier” model as we believe that the grade column is an outcome of the company’s “home model” which already uses other given columns. Moreover we see High correlation between the sub_grade and int_rate. | | | | | | | | | | | | |
| 13 | home_ownership | Clean Free | <pre>table(loan_2019_All_clean\$home_ownership)</pre> <table><thead><tr><th>ANY MORTGAGE</th><th>OWN</th><th>RENT</th></tr></thead><tbody><tr><td>59 124577</td><td>34004</td><td>110943</td></tr></tbody></table> | ANY MORTGAGE | OWN | RENT | 59 124577 | 34004 | 110943 | | | | | | |
| ANY MORTGAGE | OWN | RENT | | | | | | | | | | | | | |
| 59 124577 | 34004 | 110943 | | | | | | | | | | | | | |
| 14 | annual_inc | Clean Free - for now | Maybe dealing with outliers in the future: <pre>{r} summary(loan_2019_All_clean\$annual_inc)</pre> <table><thead><tr><th>Min.</th><th>1st Qu.</th><th>Median</th><th>Mean</th><th>3rd Qu.</th><th>Max.</th></tr></thead><tbody><tr><td>600</td><td>45000</td><td>65000</td><td>76856</td><td>90000</td><td>9573072</td></tr></tbody></table> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 600 | 45000 | 65000 | 76856 | 90000 | 9573072 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | |
| 600 | 45000 | 65000 | 76856 | 90000 | 9573072 | | | | | | | | | | |
| 16 | issued_d | 2 | We Changed the column’s type from character to date. ** This feature is important only for calculating the realized return. It won’t be necessary in the model. | | | | | | | | | | | | |
| 17 | loan_status | 5 | We filtered loans so that we will have only loans that were Fully Paid or Charged Off. * The column Can be a Target Variable | | | | | | | | | | | | |
| 21 | purpose | 3 + 4 | Using the column “title” we altered all the loans that had the value “other” in the purpose column and at the same time had a value in the column “title” that wasn’t also “other” or NA. Dealing with Outliers: We altered one loan with the value “wedding” into “other”. | | | | | | | | | | | | |
| 24 | addr_state | Clean Free | | | | | | | | | | | | | |
| 25 | dti | 1 | 35 loans with NA value in this column were filtered | | | | | | | | | | | | |
| 26 | delinq_2yrs | Clean Free | | | | | | | | | | | | | |
| 27 | earliest_cr_line | 6 | We built a new column using this column: We calculated how many years passed from the day the borrower opened his first credit line to 2016. | | | | | | | | | | | | |

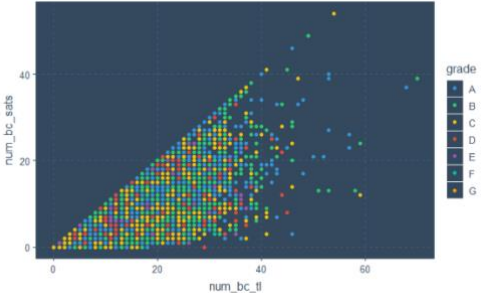
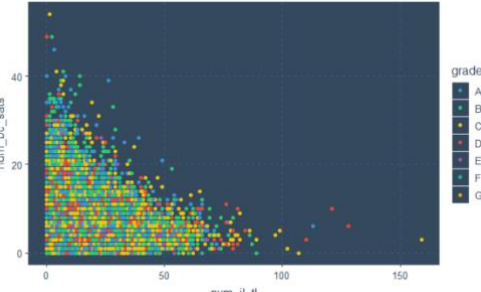
| | | | | | | | | | | | | | | | |
|---------------------|---------------------|----------------------|--|---------------------|--------------|----------------------|------|---------|-------|-----------|----------|-------|--------|-----------|----------|
| | | | <div>Scatter plot of int_rate X The new column:</div> <div></div> <div>we can see that as the number of years is smaller, so the grade increases (most of the loans with grade G are of people with little amount of years).</div> | | | | | | | | | | | | |
| 29 | fico_range_high | Clean Free | | | | | | | | | | | | | |
| 30 | inq_last_6mths | Clean Free | | | | | | | | | | | | | |
| 33 | open_acc | Clean Free | | | | | | | | | | | | | |
| 34 | pub_rec | Clean Free | | | | | | | | | | | | | |
| 35 | revol_bal | Clean Free | | | | | | | | | | | | | |
| 36 | revol_util | 2 + 1 | <div>Remove the % sign and change the column type to float.</div> <div>There are 179 NAs values - for now we will replace them with the mean value of the column, this is because after exploring the scatter plot of realized return X revol_util we found out that there isn't a distinctive correlation between the 2 variables, thus we prefer to “harm” the correlation by replacing NAs with the mean value instead of removing the rows (and losing information)</div> <div>The scatter plot of realized return X revol_util:</div> <div></div> | | | | | | | | | | | | |
| 37 | total_acc | Clean Free | <div>Maybe Dealing with outliers in the future</div> <div><pre>summary(loan_2019_All_clean\$total_acc)</pre></div> <div><table><tr><td>Min.</td><td>1st Qu.</td><td>Median</td><td>Mean</td><td>3rd Qu.</td><td>Max.</td></tr><tr><td>2.00</td><td>15.00</td><td>22.00</td><td>24.14</td><td>31.00</td><td>176.00</td></tr></table></div> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 2.00 | 15.00 | 22.00 | 24.14 | 31.00 | 176.00 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | |
| 2.00 | 15.00 | 22.00 | 24.14 | 31.00 | 176.00 | | | | | | | | | | |
| 38 | initial_list_status | Clean Free | <div><table><tr><td>initial_list_status</td><td>num_of_loans</td><td>mean_realized_return</td><td>SD</td></tr><tr><td>f</td><td>69586</td><td>0.4590696</td><td>9.414004</td></tr><tr><td>w</td><td>199962</td><td>1.2803619</td><td>7.899687</td></tr></table></div> <div>Density plot of Realized return by the two different categories of initial_list_status:</div> | initial_list_status | num_of_loans | mean_realized_return | SD | f | 69586 | 0.4590696 | 9.414004 | w | 199962 | 1.2803619 | 7.899687 |
| initial_list_status | num_of_loans | mean_realized_return | SD | | | | | | | | | | | | |
| f | 69586 | 0.4590696 | 9.414004 | | | | | | | | | | | | |
| w | 199962 | 1.2803619 | 7.899687 | | | | | | | | | | | | |

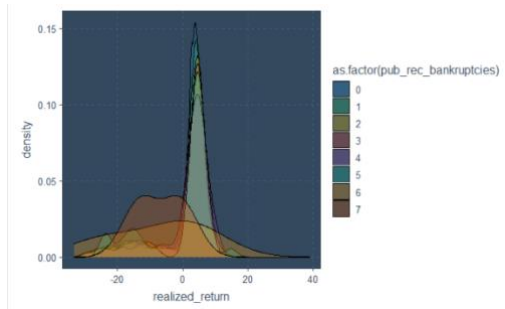
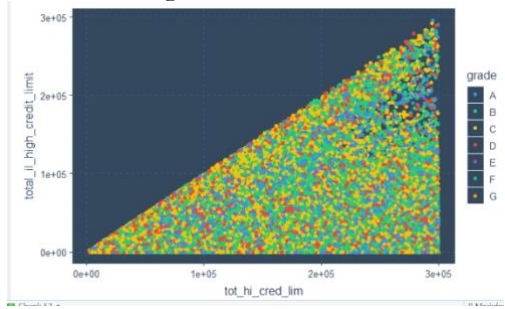
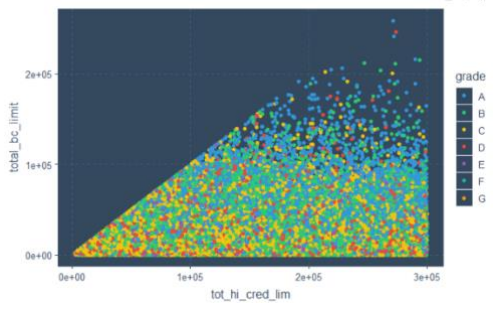
| | | |  | | | | | | | | | | | | | | | | | | | | | | | | |
|----------------------------|----------------------------|------------|---|----------------------------|-----------|--------|--------|---|------|---|-----|---|----|---|----|---|---|---|---|---|---|---|---|---|---|----|---|
| 42 | total_pymnt_in_v | ----- | Can be a Target column | | | | | | | | | | | | | | | | | | | | | | | | |
| 48 | last_pymnt_d | ----- | <p>Relevant to calculating M = the loan’s actual amount of time - meaning this is relevant for calculating the realized return. For now, we didn’t use this column to calculate the realized return, If we decide to use this column to calculate the realized return, we will need to filter 406 loans with NA value.</p> <p>Otherwise we will filter the column, thus the NA values will not be relevant.</p> | | | | | | | | | | | | | | | | | | | | | | | | |
| 54 | collections_12_mths_ex_med | Clean Free | <p>Scatter plot of realized return VS collection_12_mths_ex_med</p>  <p>It seems that if this column value is higher than 4, there is higher chance for Positive realized return</p> <table><tr><th>collections_12_mths_ex_med</th><th>count</th></tr><tr><td>0</td><td>264004</td></tr><tr><td>1</td><td>5051</td></tr><tr><td>2</td><td>335</td></tr><tr><td>3</td><td>39</td></tr><tr><td>4</td><td>22</td></tr><tr><td>5</td><td>5</td></tr><tr><td>6</td><td>4</td></tr><tr><td>7</td><td>3</td></tr><tr><td>8</td><td>1</td></tr><tr><td>9</td><td>1</td></tr><tr><td>10</td><td>1</td></tr></table> | collections_12_mths_ex_med | count | 0 | 264004 | 1 | 5051 | 2 | 335 | 3 | 39 | 4 | 22 | 5 | 5 | 6 | 4 | 7 | 3 | 8 | 1 | 9 | 1 | 10 | 1 |
| collections_12_mths_ex_med | count | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 264004 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 5051 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 335 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 39 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 22 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 4 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 57 | application_type | Clean Free | <p>We may wish to filter this column in the following step as we notices that the distribution of the realized returns for loans with application type = Individuals and for application_type = join app, is almost identical:</p>  <p>Moreover, only small amount of the loans are of type</p> <table><tr><th>Individual</th><th>Joint App</th></tr><tr><td>265364</td><td>4104</td></tr></table> <p>“Joint”:</p> | Individual | Joint App | 265364 | 4104 | | | | | | | | | | | | | | | | | | | | |
| Individual | Joint App | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 265364 | 4104 | | | | | | | | | | | | | | | | | | | | | | | | | | |

| 61 | acc_now_delinq | Clean Free | <p>Density plot of realized return by different values of acc_not_delinq:</p>  | | | | | | | | | | | | |
|------|----------------|------------|--|---------|---------|--------|------|---------|------|---|-------|-------|--------|--------|---------|
| 63 | tot_cur_bal | Clean Free | <p>Scatter plot of realized return VS tot_cur_bal</p>  <p>We can see that as the tot_cur_bal rises, so the realized tends to be positive.</p> <p>Maybe deal with Outliers in the future:</p> <table><tr><th>Min.</th><th>1st Qu.</th><th>Median</th><th>Mean</th><th>3rd Qu.</th><th>Max.</th></tr><tr><td>0</td><td>27351</td><td>70048</td><td>134542</td><td>198141</td><td>5445012</td></tr></table> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 27351 | 70048 | 134542 | 198141 | 5445012 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | |
| 0 | 27351 | 70048 | 134542 | 198141 | 5445012 | | | | | | | | | | |
| 64 | open_acc_6m | 1 | There are 44 NA's . We decided to filter them | | | | | | | | | | | | |
| 67 | open_il_24m | Clean Free | <p>Density plot of realized return by different values of open_il_24m:</p>  <p>(There were 44 NA values, However after filtering NAs in column: open_acc_6m, there were no more NAs in this column).</p> | | | | | | | | | | | | |
| 69 | total_bal_il | Clean Free | <p>Maybe dealing with outliers in the future</p> <table><tr><th>Min.</th><th>1st Qu.</th><th>Median</th><th>Mean</th><th>3rd Qu.</th><th>Max.</th></tr><tr><td>0</td><td>8592</td><td>22329</td><td>33718</td><td>43685</td><td>1547285</td></tr></table> <p>Scatter plot of realized return VS tot_cur_bal</p> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 8592 | 22329 | 33718 | 43685 | 1547285 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | |
| 0 | 8592 | 22329 | 33718 | 43685 | 1547285 | | | | | | | | | | |

| | | | | | | | | | | | | | | | |
|------|--------------------------|------------|--|---------|---------|--------|------|---------|------|---|------|------|------|------|--------|
| | | |  <p>As the total_bal_il increases, so there are more loans with positive realized return:</p> <p>Moreover we noticed a clear correlation between total_bal_il and total_bal_ex_mort:</p>  <p>The total_bal_il is always equal or smaller than the total_bal_ex_mort</p> | | | | | | | | | | | | |
| 73 | max_bal_bc | Clean Free | <p>Maybe dealing with outliers in the future:</p> <table><tr><td>Min.</td><td>1st Qu.</td><td>Median</td><td>Mean</td><td>3rd Qu.</td><td>Max.</td></tr><tr><td>0</td><td>2163</td><td>4039</td><td>5443</td><td>6968</td><td>776843</td></tr></table> <p>There are only 12 loans with max_bal_bc > 100,000.</p> <p>If we filter those 12 rows and plot a scatter plot of revol_bal vs max_bal_bc:</p>  | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 2163 | 4039 | 5443 | 6968 | 776843 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | |
| 0 | 2163 | 4039 | 5443 | 6968 | 776843 | | | | | | | | | | |
| 75 | total_rev_hi_lim | Clean Free | | | | | | | | | | | | | |
| 76 | inq_fi | Clean Free | | | | | | | | | | | | | |
| 78 | inq_last_12m | Clean Free | | | | | | | | | | | | | |
| 83 | chargeoff_within_12_mths | Clean Free | | | | | | | | | | | | | |
| 84 | delinq_amnt | Clean Free | | | | | | | | | | | | | |
| 86 | mo_sin_old_rev_tl_op | Clean Free | | | | | | | | | | | | | |

| | | | |
|----|-----------------------|------------|--|
| 87 | mo_sin_rcnt_rev_tl_op | Clean Free | Min. 1st Qu. Median Mean 3rd Qu. Max. 0.00 4.00 8.00 12.94 16.00 327.00 |
| 88 | mo_sin_rcnt_tl | Clean Free | <p>Scatter plot of mo_sin_rcnt_rev_tl_op VS mo_sin_rcnt_tl :</p>  <p>We can see that the column: mo_sin_rcnt_rev_tl_op always holds a value that is smaller or equal to mo_sin_rcnt_tl</p> |
| 89 | mort_acc | Clean Free | <p>Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 0.000 1.000 1.479 2.000 51.000</p> <p>It doesn't seem that different amount of mort accounts (between 0 to 7) influence the realized return distribution.</p>  <p>The number of mort account between 8-14 seems to influence the distribution of the realized return more:</p>  |
| 94 | num_accts_ever_120_pd | Clean Free | <p>Min. 1st Qu. Median Mean 3rd Qu. Max. 0.0000 0.0000 0.0000 0.5638 1.0000 38.0000</p> <p>We may use feature engineering - create a new column which will be the quotient of num_accts_ever_120_pd with the total number of accounts (new column we wish to create).</p> |
| 95 | num_actv_bc_tl | Clean Free | <p>Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 2.000 3.000 3.602 5.000 35.000</p> |
| 97 | num_bc_sats | Clean Free | <p>Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 3.000 4.000 4.694 6.000 54.000</p> <p>We may use feature engineering - create a new column which will be the quotient of num_bc_sats with the total number of accounts (new column we wish to create).</p> |

| | | | |
|-----|--------------------|------------|--|
| 98 | num_bc_tl | Clean Free | <div> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.00 4.00 7.00 7.58 10.00 70.00 </div> <div> <p>Scatter plot of num_bc_sats VS num_bc_tl:</p>  </div> </div> <p>We can see that as the num_bc_tl increases (the total number of bank accounts) so as the number of satisfactory bank accounts increases.</p> |
| 99 | num_il_tl | Clean Free | <div> <div> <p>Scatter plot of num_bc_sats (col 97) VS num_il_tl:</p>  </div> </div> <p>It seems that as the number of installment accounts increases, so the number of satisfactory bank accounts tends to decrease.</p> <p>*We may wish to deal with outliers in the future:</p> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 3.000 6.000 8.422 11.000 159.000 </div> |
| 100 | num_op_rev_tl | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 5.000 7.000 8.216 10.000 79.000 </div> |
| 101 | num_rev_accts | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 2.00 8.00 12.00 13.99 18.00 104.00 </div> |
| 103 | num_sats | Clean Free | <p>We may use feature engineering - create a new column which will be the quotient of num_sats with the total number of accounts (new column we wish to create). This will provide us with a column that represents the % of accounts that are satisfactory.</p> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.00 8.00 11.00 11.62 14.00 80.00 </div> <p>As there are many “outliers”, using feature engineering to create a % column can overcome this problem.</p> |
| 106 | num_tl_90g_dpd_24m | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.00000 0.00000 0.00000 0.09676 0.00000 22.00000 </div> |
| 107 | num_tl_op_past_12m | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 1.000 2.000 2.273 3.000 32.000 </div> |

| | | | |
|-----|----------------------|------------|--|
| 110 | pub_rec_bankruptcies | Clean Free | <div> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.0000 0.0000 0.0000 0.1483 0.0000 9.0000 </div> <div> 0 1 2 3 4 5 6 7 8 9 233355 33180 2183 511 133 42 13 2 4 1 </div> </div> <p>Density plot of realized return by different values of pub_rec_bankruptcies:</p>  |
| 111 | tax_liens | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.00000 0.00000 0.00000 0.07348 0.00000 61.00000 </div> |
| 112 | tot_hi_cred_lim | Clean Free | <p>*We may wish to deal with outliers in the future:</p> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 2500 47518 102282 168970 240741 9999999 </div> <p>Scatter plot of total_il_high_credit_limit VS tot_hi_cred_lim while filtering tot_hi_cred_lim<300000:</p>  <p>It clearly that as the tot_hi_cred_lim increases so as the total_il_high_credit_limit</p> |
| 113 | total_bal_ex_mort | Clean Free | <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0.000 3.000 5.000 7.769 10.000 289.000 </div> |
| 114 | total_bc_limit | 4 | <p>Scatter plot of total_bc_limit VS tot_hi_cred_lim (column 112):</p>  <p>We can see that the total bankcard limit (y) is always smaller or equal to the tot_hi_cred_lim</p> <p>Dealing with outliers: for now we filtered loans with total_bc_limit that are higher than 1,000,000 (One loan).</p> <p>This is the summary table before filtering:</p> <div> Min. 1st Qu. Median Mean 3rd Qu. Max. 0 7500 14800 21550 27800 1105500 </div> |

| | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------|----------------------------|--------|--|---------|---------|--------|------|---------|------|---|-------|-------|-------|-------|---------|------|---------|--------|------|---------|------|---|-------|-------|-------|-------|--------|
| | | | <p>This is the summary table after filtering:</p> <table><tr><td>Min.</td><td>1st Qu.</td><td>Median</td><td>Mean</td><td>3rd Qu.</td><td>Max.</td></tr><tr><td>0</td><td>7500</td><td>14800</td><td>21546</td><td>27800</td><td>616000</td></tr></table> <p>*We may wish to further deal with outliers in the future:</p> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 7500 | 14800 | 21546 | 27800 | 616000 | | | | | | | | | | | | |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 7500 | 14800 | 21546 | 27800 | 616000 | | | | | | | | | | | | | | | | | | | | | | |
| 115 | total_il_high_credit_limit | 4 | <p>Dealing with outliers: for now we filtered loans with total_il_high_credit_limit that are higher than 1,000,000 (3 loans).</p> <p>There are 3 loans with total_il_high_credit_limit that is higher than 1,000,000 (and the next highest value is 840,527)</p> <p>This is the summary table before filtering:</p> <table><tr><td>Min.</td><td>1st Qu.</td><td>Median</td><td>Mean</td><td>3rd Qu.</td><td>Max.</td></tr><tr><td>0</td><td>15000</td><td>32000</td><td>42831</td><td>57289</td><td>2000000</td></tr></table> <p>This is the summary table after filtering:</p> <table><tr><td>Min.</td><td>1st Qu.</td><td>Median</td><td>Mean</td><td>3rd Qu.</td><td>Max.</td></tr><tr><td>0</td><td>15000</td><td>32000</td><td>42816</td><td>57287</td><td>840527</td></tr></table> | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 15000 | 32000 | 42831 | 57289 | 2000000 | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | 0 | 15000 | 32000 | 42816 | 57287 | 840527 |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 15000 | 32000 | 42831 | 57289 | 2000000 | | | | | | | | | | | | | | | | | | | | | | |
| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 15000 | 32000 | 42816 | 57287 | 840527 | | | | | | | | | | | | | | | | | | | | | | |

Table 3: Matching column & NAs

In this table we will present the results of the function *match_cols* in R rmd file / *matching_columns* in Python and the function *NAs_count* in R.

The function *match_cols* in R and / *matching_columns* in Python gets as input a table from 2018 and a table from 2019, arranges each by the id column and counts how many rows match between the two tables and the matching magnitude in %. For instance, a column that has 100% match between 2018 and 2019 is a column that all of its values in 2018 are identical to the corresponding rows in 2019. Otherwise if the column has a match that is lower than 100% - some rows in the column in 2018 hold different information from the corresponding row in 2019. This can indicate that the column is a “leakage” / “varying” column and thus cannot be used. Otherwise it can be a “good” column that some of its values need addressing (typos / specific problems)

The third column Matching Rows: presents the matching % of the rows in 2018 to the loans in 2019.

The function *NAs_count* in R gets as input a table and counts how many NA values each column has and the proportion of NAs out of all the loans.

We used this function on the filtered data - the 269,583 loans that were left after filtering loans with a term that is not 36 month and loan_status that is not “Fully Paid” or “Charged Off”.

We wished to understand how many NAs values each column hold.

The fourth column presents the Number of NAs in the given column and the fifth column presents the percent of NAs out of all the loans.

| Column Number | Column Name | Matching Rows (between 2018 to 2019) | NA's (2019) out of 269,583 | NA's % |
|---------------|-------------|--------------------------------------|----------------------------|--------|
| 1 | id | 100% | 0 | 0% |
| 2 | member_id | 0% | 0 | 0% |

| | | | | |
|----|------------------------|--------|---------|--------|
| 3 | loan_amnt | 100% | 0 | 0% |
| 4 | funded_amnt | 100% | 0 | 0% |
| 5 | funded_amnt_inv | 100% | 0 | 0% |
| 6 | term | 100% | 0 | 0% |
| 7 | int_rate | 99.98% | 0 | 0% |
| 8 | installment | 99.98% | 0 | 0% |
| 9 | grade | 100% | 0 | 0% |
| 10 | sub_grade | 100% | 0 | 0% |
| 11 | emp_title | 93.44% | 19,525 | 7.24% |
| 12 | emp_length | 93.51% | 19,395 | 7.19% |
| 13 | home_ownership | 100% | 0 | 0% |
| 14 | annual_inc | 100% | 0 | 0% |
| 15 | verification_status | 98.38% | 0 | 0% |
| 16 | issue_d | 53.35% | 0 | 0% |
| 17 | loan_status | 61.36% | 0 | 0% |
| 18 | pymnt_plan | 99.94% | 0 | 0% |
| 19 | url | 100% | 0 | 0% |
| 20 | desc | 00.01% | 269,573 | 99.99% |
| 21 | purpose | 100% | 0 | 0% |
| 22 | title | 94.67% | 14,615 | 5.42% |
| 23 | zip_code | 100% | 0 | 0% |
| 24 | addr_state | 100% | 0 | 0% |
| 25 | dti | 99.99% | 35 | 0.01% |
| 26 | delinq_2yrs | 100% | 0 | 0% |
| 27 | earliest_cr_line | 0% | 0 | 0% |
| 28 | fico_range_low | 100% | 0 | 0% |
| 29 | fico_range_high | 100% | 0 | 0% |
| 30 | inq_last_6mths | 1% | 0 | 0% |
| 31 | mths_since_last_delinq | 52.85% | 127,175 | 47.17% |
| 32 | mths_since_last_record | 19.09% | 216,551 | 80.33% |

| | | | | |
|----|-----------------------------|--------|---------|--------|
| 33 | open_acc | 100% | 0 | 0% |
| 34 | pub_rec | 100% | 0 | 0% |
| 35 | revol_bal | 100% | 0 | 0% |
| 36 | revol_util | 99.94% | 179 | 0.07% |
| 37 | total_acc | 100% | 0 | 0% |
| 38 | initial_list_status | 100% | 0 | 0% |
| 39 | out_prncp | 39.96% | 0 | 0% |
| 40 | out_prncp_inv | 39.96% | 0 | 0% |
| 41 | total_pymnt | 37.47% | 0 | 0% |
| 42 | total_pymnt_inv | 37.47% | 0 | 0% |
| 43 | total_rec_prncp | 42.28% | 0 | 0% |
| 44 | total_rec_int | 42.1% | 0 | 0% |
| 45 | total_rec_late_fee | 97% | 0 | 0% |
| 46 | recoveries | 93.13% | 0 | 0% |
| 47 | collection_recovery_fee | 93.13% | 0 | 0% |
| 48 | last_pymnt_d | 24.93% | 406 | 0.15% |
| 49 | last_pymnt_amnt | 66.96% | 0 | 0% |
| 50 | next_pymnt_d | 0% | 269,583 | 100% |
| 51 | last_credit_pull_d | 9.72% | 8 | 0.003% |
| 52 | last_fico_range_high | 21.92% | 0 | 0% |
| 53 | last_fico_range_low | 21.92% | 0 | 0% |
| 54 | collections_12_mths_ex_med | 100% | 0 | 0% |
| 55 | mths_since_last_major_derog | 28.91% | 190,451 | 70.65% |
| 56 | policy_code | 100% | 0 | 0% |
| 57 | application_type | 100% | 0 | 0% |
| 58 | annual_inc_joint | 2.02% | 265,442 | 98.46% |
| 59 | dti_joint | 2.02% | 265,442 | 98.46% |
| 60 | verification_status_joint | 2.02% | 265,442 | 98.46% |
| 61 | acc_now_delinq | 100% | 0 | 0% |
| 62 | tot_coll_amt | 100% | 0 | 0% |

| | | | | |
|----|--------------------------|--------|---------|--------|
| 63 | tot_cur_bal | 100% | 0 | 0% |
| 64 | open_acc_6m | 99.99% | 44 | 0.02% |
| 65 | open_act_il | 99.99% | 44 | 0.02% |
| 66 | open_il_12m | 99.99% | 44 | 0.02% |
| 67 | open_il_24m | 99.99% | 44 | 0.02% |
| 68 | mths_since_rcnt_il | 97.29% | 8,149 | 3.02% |
| 69 | total_bal_il | 99.99% | 44 | 0.02% |
| 70 | il_util | 86.78% | 37,559 | 13.93% |
| 71 | open_rv_12m | 99.99% | 44 | 0.02% |
| 72 | open_rv_24m | 99.99% | 44 | 0.02% |
| 73 | max_bal_bc | 99.99% | 44 | 0.02% |
| 74 | all_util | 99.98% | 60 | 0.02% |
| 75 | total_rev_hi_lim | 100% | 0 | 0% |
| 76 | inq_fi | 99.99% | 44 | 0.02% |
| 77 | total_cu_tl | 99.99% | 44 | 0.02% |
| 78 | inq_last_12m | 99.99% | 44 | 0.02% |
| 79 | acc_open_past_24mths | 100% | 0 | 0% |
| 80 | avg_cur_bal | 100% | 0 | 0% |
| 81 | bc_open_to_buy | 98.93% | 3,104 | 1.15% |
| 82 | bc_util | 98.89% | 3,240 | 1.2% |
| 83 | chargeoff_within_12_mths | 100% | 0 | 0% |
| 84 | delinq_amnt | 100% | 0 | 0% |
| 85 | mo_sin_old_il_acct | 97.31% | 8,110 | 3% |
| 86 | mo_sin_old_rev_tl_op | 100% | 0 | 0% |
| 87 | mo_sin_rcnt_rev_tl_op | 100% | 0 | 0% |
| 88 | mo_sin_rcnt_tl | 100% | 0 | 0% |
| 89 | mort_acc | 100% | 0 | 0% |
| 90 | mths_since_recent_bc | 98.99% | 2,921 | 1.08% |
| 91 | mths_since_recent_bc_dlq | 25.58% | 200,380 | 74.33% |
| 92 | mths_since_recent_inq | 89.53% | 27,778 | 10.3% |

| | | | | |
|-----|--------------------------------|--------|---------|--------|
| 93 | mths_since_recent_revol_delinq | 36.27% | 171,477 | 63.61% |
| 94 | num_accts_ever_120_pd | 100% | 0 | 0% |
| 95 | num_actv_bc_tl | 100% | 0 | 0% |
| 96 | num_actv_rev_tl | 100% | 0 | 0% |
| 97 | num_bc_sats | 100% | 0 | 0% |
| 98 | num_bc_tl | 100% | 0 | 0% |
| 99 | num_il_tl | 100% | 0 | 0% |
| 100 | num_op_rev_tl | 100% | 0 | 0% |
| 101 | num_rev_accts | 100% | 0 | 0% |
| 102 | num_rev_tl_bal_gt_0 | 100% | 0 | 0% |
| 103 | num_sats | 100% | 0 | 0% |
| 104 | num_tl_120dpd_2m | 95.08% | 12,330 | 4.57% |
| 105 | num_tl_30dpd | 100% | 0 | 0% |
| 106 | num_tl_90g_dpd_24m | 100% | 0 | 0% |
| 107 | num_tl_op_past_12m | 100% | 0 | 0% |
| 108 | pct_tl_nvr_dlq | 100% | 0 | 0% |
| 109 | percent_bc_gt_75 | 98.93% | 3,127 | 1.16% |
| 110 | pub_rec_bankruptcies | 100% | 0 | 0% |
| 111 | tax_liens | 100% | 0 | 0% |
| 112 | tot_hi_cred_lim | 100% | 0 | 0% |
| 113 | total_bal_ex_mort | 100% | 0 | 0% |
| 114 | total_bc_limit | 100% | 0 | 0% |
| 115 | total_il_high_credit_limit | 100% | 0 | 0% |
| 116 | revol_bal_joint | 0% | 269,583 | 100% |
| 117 | sec_app_fico_range_low | 0% | 269,583 | 100% |
| 118 | sec_app_fico_range_high | 0% | 269,583 | 100% |
| 119 | sec_app_earliest_cr_line | 0% | 269,583 | 100% |
| 120 | sec_app_inq_last_6mths | 0% | 269,583 | 100% |
| 121 | sec_app_mort_acc | 0% | 269,583 | 100% |
| 122 | sec_app_open_acc | 0% | 269,583 | 100% |

| | | | | |
|-----|--|--------|---------|--------|
| 123 | sec_app_revol_util | 0% | 269,583 | 100% |
| 124 | sec_app_open_act_il | 0% | 269,583 | 100% |
| 125 | sec_app_num_rev_accts | 0% | 269,583 | 100% |
| 126 | sec_app_chargeoff_within_12_mths | 0% | 269,583 | 100% |
| 127 | sec_app_collections_12_mths_ex_me d | 0% | 269,583 | 100% |
| 128 | sec_app_mths_since_last_major_dero g | 0% | 269,583 | 100% |
| 129 | hardship_flag | 99.91% | 0 | 0% |
| 130 | hardship_type | 0.74% | 267,415 | 99.2% |
| 131 | hardship_reason | 0.74% | 267,415 | 99.2% |
| 132 | hardship_status | 0.67% | 267,415 | 99.2% |
| 133 | deferral_term | 0.74% | 267,415 | 99.2% |
| 134 | hardship_amount | 0.74% | 267,415 | 99.2% |
| 135 | hardship_start_date | 0.4% | 267,415 | 99.2% |
| 136 | hardship_end_date | 0.39% | 267,415 | 99.2% |
| 137 | payment_plan_start_date | 0.4% | 267,415 | 99.2% |
| 138 | hardship_length | 0.74% | 267,415 | 99.2% |
| 139 | hardship_dpd | 0.74% | 267,415 | 99.2% |
| 140 | hardship_loan_status | 0.74% | 267,415 | 99.2% |
| 141 | orig_projected_additional_accrued_int erest | 0.56% | 267,415 | 99.2% |
| 142 | hardship_payoff_balance_amount | 0.74% | 267,415 | 99.2% |
| 143 | hardship_last_payment_amount | 0.74% | 267,415 | 99.2% |
| 144 | disbursement_method | 0% | 0 | 0% |
| 145 | debt_settlement_flag | 98.17% | 0 | 0% |
| 146 | debt_settlement_flag_date | 0.29% | 261,031 | 96.83% |
| 147 | settlement_status | 0.4% | 261,031 | 96.83% |
| 148 | settlement_date | 0.69% | 261,031 | 96.83% |
| 149 | settlement_amount | 1.03% | 261,031 | 96.83% |
| 150 | settlement_percentage | 1.04% | 261,031 | 96.83% |

| | | | | |
|-----|-----------------|-------|---------|--------|
| 151 | settlement_term | 1.05% | 261,031 | 96.83% |
|-----|-----------------|-------|---------|--------|