IDS 572 Assigment 1 Part A

Lauren Sansone, Joshua Pollack, Lina Quiceno Bejarano

Question 2a - Part(I)

What is the proportion of defaults ('charged off' vs 'fully paid' loans) in the data?

The total number of loans is 81,022 loans. The proportion of fully paid loans is 69,195 and charged off loans are 11,827. Charged off loans represent 14.6% of the total amount of loans. It is expected that the vast majority of loans funded would be fully paid (85.4% in this case).

```
## # A tibble: 2 x 3
##   loan_status nLoans prctTot
## * <chr>        <int>   <dbl>
## 1 Charged Off  11827    14.6
## 2 Fully Paid   69195    85.4
```

How does default rate vary with loan grade?

Exploring a table of percent of charged off loans and fully paid loans by total loans in each grade, the grade with the highest percentage of charged off loans is grade G, with 41.8% of loans charged off. This is expected as loan grade G is the riskiest grade category. Default rate increases as loan grade decreases nearly linearly. This is expected as the loans that are considered riskier by loan grade have a greater percentage of charged off grades.

```
## # A tibble: 14 x 4
## # Groups:   grade [7]
##    grade loan_status nLoans prctTot
##    <chr> <chr>        <int>   <dbl>
##  1 A     Charged Off   1108    5.43
##  2 A     Fully Paid   19294   94.6
##  3 B     Charged Off   2682   11.5
##  4 B     Fully Paid   20717   88.5
##  5 C     Charged Off   4116   18.2
##  6 C     Fully Paid   18461   81.8
##  7 D     Charged Off   2647   24.5
##  8 D     Fully Paid    8155   75.5
##  9 E     Charged Off   1045   32.7
## 10 E     Fully Paid    2146   67.3
## 11 F     Charged Off    191   34.1
## 12 F     Fully Paid     369   65.9
## 13 G     Charged Off     38   41.8
## 14 G     Fully Paid      53   58.2
```

```
## # A tibble: 14 x 4
## # Groups:   loan_status [2]
##    loan_status grade nLoans prctTot
##    <chr>       <chr>  <int>   <dbl>
##  1 Charged Off A       1108  9.37
##  2 Charged Off B       2682 22.7
```

```
##   3 Charged Off C       4116 34.8
##   4 Charged Off D       2647 22.4
##   5 Charged Off E       1045  8.84
##   6 Charged Off F        191  1.61
##   7 Charged Off G         38  0.321
##   8 Fully Paid  A      19294 27.9
##   9 Fully Paid  B      20717 29.9
##  10 Fully Paid  C      18461 26.7
##  11 Fully Paid  D       8155 11.8
##  12 Fully Paid  E       2146  3.10
##  13 Fully Paid  F        369  0.533
##  14 Fully Paid  G         53  0.0766
```

Does it vary with sub-grade? And is this what you would expect, and why?

The proportion of loans that are charged off does vary by subgrade. The percent of loans that are charged off generally increase as loan grade decreases. Again, this is as expected as loans that are considered riskier by grade have a greater percentage of charged off loans.

```
## # A tibble: 70 x 4
## # Groups:   sub_grade [35]
##    sub_grade loan_status nLoans prctTot
##    <chr>     <chr>        <int>   <dbl>
##  1 A1        Charged Off     95    3.14
##  2 A1        Fully Paid    2927   96.9
##  3 A2        Charged Off    165    4.61
##  4 A2        Fully Paid    3418   95.4
##  5 A3        Charged Off    154    4.34
##  6 A3        Fully Paid    3394   95.7
##  7 A4        Charged Off    281    5.81
##  8 A4        Fully Paid    4558   94.2
##  9 A5        Charged Off    413    7.63
## 10 A5        Fully Paid    4997   92.4
## # ... with 60 more rows
```

```
## # A tibble: 70 x 4
## # Groups:   loan_status [2]
##    loan_status sub_grade nLoans prctTot
##    <chr>       <chr>      <int>   <dbl>
##  1 Charged Off A1            95   0.803
##  2 Charged Off A2           165   1.40
##  3 Charged Off A3           154   1.30
##  4 Charged Off A4           281   2.38
##  5 Charged Off A5           413   3.49
##  6 Charged Off B1           373   3.15
##  7 Charged Off B2           479   4.05
##  8 Charged Off B3           531   4.49
##  9 Charged Off B4           546   4.62
## 10 Charged Off B5           753   6.37
## # ... with 60 more rows
```

Question 2a - Part(II)

How many loans are there in each grade? And do loan amounts vary by grade?
Does interest rate for loans vary with grade, subgrade? Look at the average, standard-

deviation, min and max of interest rate by grade and subgrade. Is this what you expect, and why?

The number of loans per grade:
A = 20402; B = 23399; C = 22577; D = 10802; E = 3191; F = 560; G = 91

The majority of loans are in grade A, B and C with the least amount of loans in grade G. This is expected as it would make sense for the company to invest in less risky loans.

The average loan amount does not vary much by grade. The average range of loan amounts are between $10,000 to $14,000.

Interest rates certainly vary by grade and subgrade. Average interest rates increase as loan grades decrease both across grades and within subgrades. This is expected as higher interest rates are applied to riskier loans. It shows that Lending Club is basing interest rates given off of loan grades.

Standard deviations in groups and subgroups are small. This makes sense as interest rates are likely determined by loan grade.

Average, min and max interest rates are as expected as they follow the general pattern that interest rates increase as grade decreases.

```
## # A tibble: 7 x 2
##   grade     n
## * <chr> <int>
## 1 A     20402
## 2 B     23399
## 3 C     22577
## 4 D     10802
## 5 E      3191
## 6 F       560
## 7 G        91
```

```
## # A tibble: 7 x 2
##   grade `sum(loan_amnt)`
## * <chr>            <dbl>
## 1 A            288605200
## 2 B            291509175
## 3 C            258857100
## 4 D            131244900
## 5 E             40072800
## 6 F              5694425
## 7 G              1138325
```

```
## # A tibble: 7 x 2
##   grade `mean(loan_amnt)`
## * <chr>             <dbl>
## 1 A                14146.
## 2 B                12458.
## 3 C                11466.
## 4 D                12150.
## 5 E                12558.
## 6 F                10169.
## 7 G                12509.
```

```
## # A tibble: 7 x 2
##   grade `mean(int_rate)`
## * <chr>            <dbl>
## 1 A                 7.25
```

```
## 2 B                10.7
## 3 C                13.7
## 4 D                16.5
## 5 E                19.8
## 6 F                24.1
## 7 G                25.8
```

```
## # A tibble: 35 x 2
##    sub_grade `mean(int_rate)`
##  * <chr>                <dbl>
##  1 A1                    6.03
##  2 A2                    6.49
##  3 A3                    7.05
##  4 A4                    7.58
##  5 A5                    8.27
##  6 B1                    8.88
##  7 B2                    9.77
##  8 B3                   10.7
##  9 B4                   11.5
## 10 B5                   12.2
## 11 C1                   12.7
## 12 C2                   13.1
## 13 C3                   13.8
## 14 C4                   14.4
## 15 C5                   15.0
## 16 D1                   15.6
## 17 D2                   16.1
## 18 D3                   16.7
## 19 D4                   17.3
## 20 D5                   18.0
## 21 E1                   18.7
## 22 E2                   19.4
## 23 E3                   20.1
## 24 E4                   21.0
## 25 E5                   22.1
## 26 F1                   23.2
## 27 F2                   24.0
## 28 F3                   24.5
## 29 F4                   25.0
## 30 F5                   25.6
## 31 G1                   25.8
## 32 G2                   25.8
## 33 G3                   25.9
## 34 G4                   26.0
## 35 G5                   26.1
```

```
## # A tibble: 7 x 6
##   grade nLoans avgInterest stdInterest minInt maxInt
## * <chr>  <int>       <dbl>       <dbl>  <dbl>  <dbl>
## 1 A      20402        7.25       0.796     6    8.39
## 2 B      23399       10.7        1.22      6   12.5
## 3 C      22577       13.7        0.850     6   15.0
## 4 D      10802       16.5        0.895     6   18.2
## 5 E       3191       19.8        1.10      6   22.2
## 6 F        560       24.1        0.798    23.0  25.6
```

```
## 7 G         91       25.8       0.0593   25.8  26.1
```

```
## # A tibble: 35 x 6
##    sub_grade nLoans avgInterest stdInterest minInt maxInt
##  * <chr>      <int>       <dbl>       <dbl>  <dbl>  <dbl>
##  1 A1          3022        6.03    0.000546   6      6.03
##  2 A2          3583        6.49    0          6.49   6.49
##  3 A3          3548        7.05    0.0650     6.99   7.12
##  4 A4          4839        7.58    0.0996     7.49   7.69
##  5 A5          5410        8.27    0.0971     8.19   8.39
##  6 B1          4123        8.88    0.251      6      9.17
##  7 B2          4604        9.77    0.326      9.49  10.2
##  8 B3          4603       10.7     0.248     10.5   11.0
##  9 B4          4628       11.5     0.162      6     11.7
## 10 B5          5441       12.2     0.260      6     12.5
## 11 C1          5410       12.7     0.298     12.4   13.0
## 12 C2          5102       13.1     0.179     13.0   13.4
## 13 C3          4539       13.8     0.197      6     14.0
## 14 C4          3876       14.4     0.162      6     14.5
## 15 C5          3650       15.0     0         15.0   15.0
## 16 D1          3003       15.6     0.248      6     15.6
## 17 D2          2493       16.1     0.252      6     16.3
## 18 D3          2175       16.7     0.339      6     17.0
## 19 D4          1734       17.3     0.585      6     17.6
## 20 D5          1397       18.0     0.373      6     18.2
## 21 E1          1066       18.7     0.450      6     19.0
## 22 E2           807       19.4     0.140     19.2   19.5
## 23 E3           582       20.1     0.105     20.0   20.2
## 24 E4           423       21.0     0         21.0   21.0
## 25 E5           313       22.1     0.0798    22.0   22.2
## 26 F1           202       23.2     0.218     23.0   23.4
## 27 F2           121       24.0     0.0446    24.0   24.1
## 28 F3           105       24.5     0         24.5   24.5
## 29 F4            77       25.0     0         25.0   25.0
## 30 F5            55       25.6     0         25.6   25.6
## 31 G1            42       25.8     0         25.8   25.8
## 32 G2            27       25.8     0         25.8   25.8
## 33 G3            15       25.9     0         25.9   25.9
## 34 G4             5       26.0     0         26.0   26.0
## 35 G5             2       26.1     0         26.1   26.1
```

Question 2a - Part(III)

What are people borrowing money for (purpose)? Examine how many loans, average amounts, etc. by purpose? And within grade? Do defaults vary by purpose?

The purpose people are borrowing money include car loans, credit cards, debt consolidation, home improvement projects, house purchases, major purchases, medical, moving, renewable energy, small business, vacations and weddings.

The majority of loans are for the purpose of debt consolidation (60%) and credit cards (23.2%). Weddings have the least amount of loans, totaling only three loans.

The average dollar amount of loans vary by purpose (ranging from the lowest average of $5,872 for vacations to the highest average of $14,425 for small business).

Across all purpose categories, the fewest number of loans are in loan grades E, F and G. Except in the credit card category, the majority count of loans are in grades B, C and D (with a slightly less number of loans in grade A). The credit card category is the only purpose with the most amount of loans in loan grade A compared to the other grades.

The highest total number of defaults are in debt consolidation, which also is the greatest category for purpose of loan. The highest percentage of defaults by purpose are from the small business category, which has 22.3% of the loans charged off.

```
## # A tibble: 11 x 2
##    purpose                n
##  * <fct>              <int>
##  1 car                  719
##  2 credit_card        18780
##  3 debt_consolidation 48647
##  4 home_improvement    3942
##  5 house                254
##  6 major_purchase      1402
##  7 medical              900
##  8 moving               604
##  9 other               4523
## 10 small_business       759
## 11 vacation             492
```

```
## # A tibble: 11 x 3
##    purpose            nLoans prctTot
##  * <fct>               <int>   <dbl>
##  1 car                   719   0.887
##  2 credit_card         18780  23.2
##  3 debt_consolidation  48647  60.0
##  4 home_improvement     3942   4.87
##  5 house                 254   0.313
##  6 major_purchase       1402   1.73
##  7 medical               900   1.11
##  8 moving                604   0.745
##  9 other                4523   5.58
## 10 small_business        759   0.937
## 11 vacation              492   0.607
```

```
## # A tibble: 11 x 6
##    purpose            nLoans avgInterest avgLoanAmt defaults prctCharged_off
##  * <fct>               <int>       <dbl>      <dbl>    <int>           <dbl>
##  1 car                   719        11.8      7820.       75            10.4
##  2 credit_card         18780        10.3     13501.     2326            12.4
##  3 debt_consolidation  48647        12.1     13008.     7423            15.3
##  4 home_improvement     3942        11.8     11707.      503            12.8
##  5 house                 254        16.3     12505.       48            18.9
##  6 major_purchase       1402        12.0     10172.      220            15.7
##  7 medical               900        13.8      6981.      141            15.7
##  8 moving                604        15.5      6542.      132            21.9
##  9 other                4523        14.1      8293.      718            15.9
## 10 small_business        759        16.4     14425.      169            22.3
## 11 vacation              492        13.7      5872.       72            14.6
```

```
## # A tibble: 76 x 3
## # Groups:   grade [7]
```

```
##    grade purpose                n
##    <chr> <fct>              <int>
##  1 A     car                  194
##  2 A     credit_card         7487
##  3 A     debt_consolidation 10824
##  4 A     home_improvement    1043
##  5 A     house                  8
##  6 A     major_purchase       361
##  7 A     medical               73
##  8 A     moving                 8
##  9 A     other                340
## 10 A     small_business        26
## 11 A     vacation              38
## 12 B     car                  209
## 13 B     credit_card         6105
## 14 B     debt_consolidation 14124
## 15 B     home_improvement    1130
## 16 B     house                 29
## 17 B     major_purchase       381
## 18 B     medical              199
## 19 B     moving                63
## 20 B     other                987
## 21 B     small_business        66
## 22 B     vacation             106
## 23 C     car                  195
## 24 C     credit_card         3779
## 25 C     debt_consolidation 14419
## 26 C     home_improvement    1059
## 27 C     house                 69
## 28 C     major_purchase       377
## 29 C     medical              371
## 30 C     moving               240
## 31 C     other               1654
## 32 C     small_business       218
## 33 C     vacation             196
## 34 D     car                   79
## 35 D     credit_card         1135
## 36 D     debt_consolidation  6972
## 37 D     home_improvement     520
## 38 D     house                 78
## 39 D     major_purchase       215
## 40 D     medical              188
## 41 D     moving               205
## 42 D     other               1045
## 43 D     small_business       248
## 44 D     vacation             117
## 45 E     car                   30
## 46 E     credit_card          236
## 47 E     debt_consolidation  1967
## 48 E     home_improvement     161
## 49 E     house                 43
## 50 E     major_purchase        59
## 51 E     medical               54
## 52 E     moving                66
```

```
## 53 E     other                401
## 54 E     small_business       142
## 55 E     vacation              32
## 56 F     car                    9
## 57 F     credit_card           31
## 58 F     debt_consolidation   308
## 59 F     home_improvement      26
## 60 F     house                 18
## 61 F     major_purchase         7
## 62 F     medical               14
## 63 F     moving                20
## 64 F     other                 80
## 65 F     small_business        44
## 66 F     vacation               3
## 67 G     car                    3
## 68 G     credit_card            7
## 69 G     debt_consolidation    33
## 70 G     home_improvement       3
## 71 G     house                  9
## 72 G     major_purchase         2
## 73 G     medical                1
## 74 G     moving                 2
## 75 G     other                 16
## 76 G     small_business        15

## # A tibble: 76 x 3
## # Groups:   purpose [11]
##    purpose            grade     n
##    <fct>              <chr> <int>
##  1 car                A       194
##  2 car                B       209
##  3 car                C       195
##  4 car                D        79
##  5 car                E        30
##  6 car                F         9
##  7 car                G         3
##  8 credit_card        A      7487
##  9 credit_card        B      6105
## 10 credit_card        C      3779
## 11 credit_card        D      1135
## 12 credit_card        E       236
## 13 credit_card        F        31
## 14 credit_card        G         7
## 15 debt_consolidation A     10824
## 16 debt_consolidation B     14124
## 17 debt_consolidation C     14419
## 18 debt_consolidation D      6972
## 19 debt_consolidation E      1967
## 20 debt_consolidation F       308
## 21 debt_consolidation G        33
## 22 home_improvement   A      1043
## 23 home_improvement   B      1130
## 24 home_improvement   C      1059
## 25 home_improvement   D       520
```

```
## 26 home_improvement   E       161
## 27 home_improvement   F        26
## 28 home_improvement   G         3
## 29 house              A         8
## 30 house              B        29
## 31 house              C        69
## 32 house              D        78
## 33 house              E        43
## 34 house              F        18
## 35 house              G         9
## 36 major_purchase     A       361
## 37 major_purchase     B       381
## 38 major_purchase     C       377
## 39 major_purchase     D       215
## 40 major_purchase     E        59
## 41 major_purchase     F         7
## 42 major_purchase     G         2
## 43 medical            A        73
## 44 medical            B       199
## 45 medical            C       371
## 46 medical            D       188
## 47 medical            E        54
## 48 medical            F        14
## 49 medical            G         1
## 50 moving             A         8
## 51 moving             B        63
## 52 moving             C       240
## 53 moving             D       205
## 54 moving             E        66
## 55 moving             F        20
## 56 moving             G         2
## 57 other              A       340
## 58 other              B       987
## 59 other              C      1654
## 60 other              D      1045
## 61 other              E       401
## 62 other              F        80
## 63 other              G        16
## 64 small_business     A        26
## 65 small_business     B        66
## 66 small_business     C       218
## 67 small_business     D       248
## 68 small_business     E       142
## 69 small_business     F        44
## 70 small_business     G        15
## 71 vacation           A        38
## 72 vacation           B       106
## 73 vacation           C       196
## 74 vacation           D       117
## 75 vacation           E        32
## 76 vacation           F         3
```

```
## # A tibble: 11 x 4
##    purpose            nLoans defaults prctCharged_off
```

```
##   * <fct>              <int>    <int>           <dbl>
##   1 car                  719       75            10.4
##   2 credit_card        18780     2326            12.4
##   3 debt_consolidation 48647     7423            15.3
##   4 home_improvement    3942      503            12.8
##   5 house                254       48            18.9
##   6 major_purchase      1402      220            15.7
##   7 medical              900      141            15.7
##   8 moving               604      132            21.9
##   9 other               4523      718            15.9
##  10 small_business       759      169            22.3
##  11 vacation             492       72            14.6
```

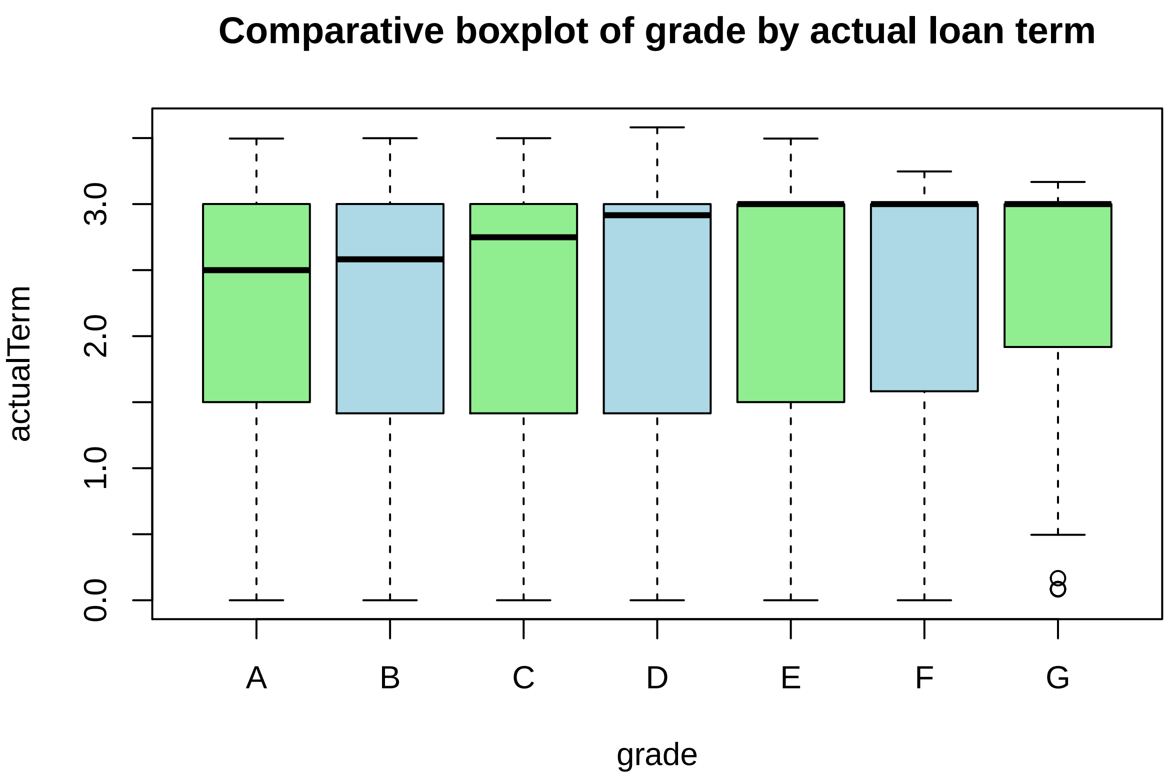
Question 2a - Part (IV)

For loans which are fully paid back, how does the time-to-full-payoff vary? For this, calculate the 'actual term' (issue-date to last-payment-date) for all loans. How does this actual-term vary by loan grade (a box-plot can help visualize this).

The time to full payoff for loans varies from 43.43 weeks (0.8 years) to 3.09 years.

The average actual payoff term varies slightly by loan grade. Grade A has the shortest payoff time of 2.22 years and the payoff time increases as grade decreases. This shows how the grade is dependent on the risk associated with each loan and chances of getting it back. This boxplot helps illustrate that loans are paid back quicker from A grades than from lower grade like D, E, & F.

```
## # A tibble: 7 x 2
##   grade `mean(actualTerm)`
## * <chr>              <dbl>
## 1 A                   2.22
## 2 B                   2.21
## 3 C                   2.22
## 4 D                   2.26
## 5 E                   2.31
## 6 F                   2.33
## 7 G                   2.46
```



Question 2a - Part (V)

Calculate the annual return. Show how you calculate the percentage annual return. Is there any return from loans which are 'charged off'? Explain. How does return from charged - off loans vary by loan grade? Compare the average return values with the average interest_rate on loans – do you notice any differences, and how do you explain this? How do returns vary by grade, and by sub-grade. If you wanted to invest in loans based on this data exploration, which loans would you invest in?

The annual percentage return is calculated by: lcdf$actualReturn <- ifelse(lcdf$actualTerm>0, ((lcdf$total_pymnt -lcdf$funded_amnt)/lcdf$funded_amnt)*(1/lcdf$actualTerm)*100, 0)

There are no returns from loans that are charged off. There is a total loss of $138,346 from all charged off loans. For charged off loans, the total payments collected are less than the funded amount, and therefore the annual return is a loss.

The total returns from charged off loans vary in each loan grade category. All charged off returns are negative which means a loss (not a return). Loan grade C has the highest dollar amount lost of $47,862.

Compared to the average interest rate on loans, average return and average interest rates generally increases as loan grade decreases. Although, in the lowest loan grade G, average return is lower than loan grade F.

The same pattern also follows within subgrades A through E; average return and average interest rates generally increase within subgrade as subgrade decreases. There is a variation in subgrades F and G: as interest rates increase, there is no pattern within the two subgrades. Subgrade G4 is the only subgrade with a negative return.

According to the data, the highest average return occurs in subgrade F4. This may be attractive for investing purposes, although F4 is a high-risk loan grade. To minimize risk, the best loans to invest in are the lowest risk loans (grade A) with the highest return (subgrade A5).

```
## # A tibble: 2 x 2
```

```
##   loan_status `sum(actualReturn)`
## * <chr>                     <dbl>
## 1 Charged Off            -138346.
## 2 Fully Paid              555728.

## # A tibble: 14 x 5
## # Groups:   loan_status [2]
##    loan_status grade nLoans totActRet avgActRet
##    <chr>       <chr>  <int>     <dbl>     <dbl>
##  1 Charged Off A       1108   -12464.    -11.2
##  2 Charged Off B       2682   -29521.    -11.0
##  3 Charged Off C       4116   -47862.    -11.6
##  4 Charged Off D       2647   -32348.    -12.2
##  5 Charged Off E       1045   -13340.    -12.8
##  6 Charged Off F        191    -2395.    -12.5
##  7 Charged Off G         38     -417.    -11.0
##  8 Fully Paid  A      19294    92768.      4.81
##  9 Fully Paid  B      20717   151607.      7.32
## 10 Fully Paid  C      18461   177225.      9.60
## 11 Fully Paid  D       8155    95940.     11.8
## 12 Fully Paid  E       2146    30720.     14.3
## 13 Fully Paid  F        369     6475.     17.5
## 14 Fully Paid  G         53      993.     18.7

## # A tibble: 7 x 3
##   grade avgActRet avgInt
## * <chr>     <dbl>  <dbl>
## 1 A          3.94   7.25
## 2 B          5.22  10.7
## 3 C          5.73  13.7
## 4 D          5.89  16.5
## 5 E          5.45  19.8
## 6 F          7.29  24.1
## 7 G          6.33  25.8

## # A tibble: 35 x 3
##    sub_grade avgActRet avgInt
##  * <chr>         <dbl>  <dbl>
##  1 A1             3.55   6.03
##  2 A2             3.53   6.49
##  3 A3             3.92   7.05
##  4 A4             4.11   7.58
##  5 A5             4.28   8.27
##  6 B1             4.45   8.88
##  7 B2             4.77   9.77
##  8 B3             5.25  10.7
##  9 B4             5.72  11.5
## 10 B5             5.72  12.2
## 11 C1             5.70  12.7
## 12 C2             5.44  13.1
## 13 C3             5.42  13.8
## 14 C4             5.90  14.4
## 15 C5             6.39  15.0
## 16 D1             5.65  15.6
## 17 D2             5.53  16.1
```

```
## 18 D3              6.38  16.7
## 19 D4              6.17  17.3
## 20 D5              5.92  18.0
## 21 E1              5.31  18.7
## 22 E2              5.28  19.4
## 23 E3              5.58  20.1
## 24 E4              5.53  21.0
## 25 E5              5.97  22.1
## 26 F1              7.54  23.2
## 27 F2              6.47  24.0
## 28 F3              8.05  24.5
## 29 F4              8.11  25.0
## 30 F5              5.55  25.6
## 31 G1              6.67  25.8
## 32 G2              7.44  25.8
## 33 G3              6.42  25.9
## 34 G4             -3.40  26.0
## 35 G5              8.03  26.1
```

Question 2a - Part(VI) derived attributes

Generate some (at least 3) new derived attributes which you think may be useful for predicting default, and explain what these are. New attributes that could be helpful in predicting default include:

1. Loan status (fully paid vs. charged off) compared to num_bc_sats, the total number of the borrower's satisfactory bankcard accounts. If the borrower has many satisfactory bankcard accounts, it may indicate lower risk of default.
2. Loan status (fully paid vs. charged off) compared to open_acc, the number of open credit lines in the borrower's credit file. If the borrower has many open credit lines, it may indicate higher risk of default.
3. Loan status (fully paid vs. charged off) compared to acc_now_delinq, The number of accounts on which the borrower is now delinquent. If the borrower has many accounts with delinquencies, it may indicate higher risk of default.

```r
#Proportion of Satisfactory Bank Card
lcdf$propSatisBankcardAccts <- ifelse(lcdf$num_bc_tl>0, lcdf$num_bc_sats/lcdf$num_bc_tl, 0)

#Length of borrower history
lcdf$earliest_cr_line<-paste(lcdf$earliest_cr_line, "-01", sep = "")
lcdf$earliest_cr_line<-parse_date_time(lcdf$earliest_cr_line, "myd")
lcdf$borrHistory <- as.duration(lcdf$earliest_cr_line %--% lcdf$issue_d  ) / dyears(1)

#Ratio of open accounts
lcdf$ratio_openAccounts <- ifelse(lcdf$total_acc>0, lcdf$open_acc/lcdf$total_acc, 0)

#Proportion of delinquent accounts
lcdf$prop_delinquent <- ifelse(lcdf$open_acc>0, lcdf$acc_now_delinq/lcdf$open_acc,0)
```

Question 2b - Missing values

Are there missing values? What is the proportion of missing values in different variables? Explain how you will handle missing values for different variables. You should consider what the variable is about, and what missing values may arise from – for example, a variable

monthsSinceLastDeliquency may have no value for someone who has not yet had a delinquency; what is a sensible value to replace the missing values in this case? Are there some variables you will exclude from your model due to missing values?

Yes, there are missing values.

We excluded some variables because they did not have any values. Initially we had 149 variables, after running our code for missing variables, we kept 89 variables.

There are two columns with missing values emp_title and last_credit_pull_d. The proportion of missing values for emp_title is 0.0630939745 and last_credit_pull_d is 0.0001481079.

The variable mths_since_last_delinq has 48% missings values. We are going to replace those values with a value higher than the max (500) because the missing values pertain to non delinquency. We are going to use this same technique for the variables: mo_sin_old_il_acct=1000, mths_since_recent_bc=1000, and mths_since_recent_inq=50.

For the next variables we are going to handle missing values with the median: revol_util, bc_open_to_buy, percent_bc_gt_75, bc_util.

```
#Drop vars with all empty values
dim(lcdf)
```

```
## [1] 81022   153
```

```
lcdf <- lcdf %>% select_if(function(x){!all(is.na(x))})
dim(lcdf)
```

```
## [1] 81022   103
```

```
#Of the columns remaining, names of columns with missing values
names(lcdf)[colSums(is.na(lcdf))>0]
```

```
##  [1] "emp_title"                    "mths_since_last_delinq"
##  [3] "mths_since_last_record"       "revol_util"
##  [5] "last_pymnt_d"                 "last_credit_pull_d"
##  [7] "mths_since_last_major_derog"  "bc_open_to_buy"
##  [9] "bc_util"                      "mo_sin_old_il_acct"
## [11] "mths_since_recent_bc"         "mths_since_recent_bc_dlq"
## [13] "mths_since_recent_inq"        "mths_since_recent_revol_delinq"
## [15] "num_tl_120dpd_2m"             "percent_bc_gt_75"
## [17] "hardship_dpd"                 "hardship_last_payment_amount"
## [19] "settlement_term"
```

```
#missing value proportions in each column
colMeans(is.na(lcdf))
```

```
##                          X1                    loan_amnt
##                0.0000000000                 0.0000000000
##                 funded_amnt              funded_amnt_inv
##                0.0000000000                 0.0000000000
##                        term                     int_rate
##                0.0000000000                 0.0000000000
##                 installment                        grade
##                0.0000000000                 0.0000000000
##                   sub_grade                    emp_title
##                0.0000000000                 0.0630939745
##                  emp_length               home_ownership
##                0.0000000000                 0.0000000000
##                  annual_inc          verification_status
```

```
##                  0.0000000000                   0.0000000000
##                       issue_d                    loan_status
##                  0.0000000000                   0.0000000000
##                    pymnt_plan                        purpose
##                  0.0000000000                   0.0000000000
##                         title                       zip_code
##                  0.0000000000                   0.0000000000
##                    addr_state                            dti
##                  0.0000000000                   0.0000000000
##                   delinq_2yrs               earliest_cr_line
##                  0.0000000000                   0.0000000000
##                inq_last_6mths         mths_since_last_delinq
##                  0.0000000000                   0.4782898472
##        mths_since_last_record                       open_acc
##                  0.8178642838                   0.0000000000
##                       pub_rec                      revol_bal
##                  0.0000000000                   0.0000000000
##                    revol_util                      total_acc
##                  0.0004196391                   0.0000000000
##           initial_list_status                      out_prncp
##                  0.0000000000                   0.0000000000
##                 out_prncp_inv                    total_pymnt
##                  0.0000000000                   0.0000000000
##               total_pymnt_inv                total_rec_prncp
##                  0.0000000000                   0.0000000000
##                 total_rec_int             total_rec_late_fee
##                  0.0000000000                   0.0000000000
##                    recoveries        collection_recovery_fee
##                  0.0000000000                   0.0000000000
##                  last_pymnt_d                last_pymnt_amnt
##                  0.0005554047                   0.0000000000
##            last_credit_pull_d     collections_12_mths_ex_med
##                  0.0001481079                   0.0000000000
##   mths_since_last_major_derog                    policy_code
##                  0.7033521760                   0.0000000000
##              application_type                  acc_now_delinq
##                  0.0000000000                   0.0000000000
##                  tot_coll_amt                    tot_cur_bal
##                  0.0000000000                   0.0000000000
##              total_rev_hi_lim           acc_open_past_24mths
##                  0.0000000000                   0.0000000000
##                   avg_cur_bal                 bc_open_to_buy
##                  0.0000000000                   0.0120584533
##                       bc_util       chargeoff_within_12_mths
##                  0.0126879119                   0.0000000000
##                   delinq_amnt             mo_sin_old_il_acct
##                  0.0000000000                   0.0375330157
##          mo_sin_old_rev_tl_op           mo_sin_rcnt_rev_tl_op
##                  0.0000000000                   0.0000000000
##                mo_sin_rcnt_tl                       mort_acc
##                  0.0000000000                   0.0000000000
##          mths_since_recent_bc        mths_since_recent_bc_dlq
##                  0.0112068327                   0.7287526845
##         mths_since_recent_inq mths_since_recent_revol_delinq
```

```
##                     0.1014786107                     0.6293352423
##           num_accts_ever_120_pd                  num_actv_bc_tl
##                     0.0000000000                     0.0000000000
##                  num_actv_rev_tl                      num_bc_sats
##                     0.0000000000                     0.0000000000
##                        num_bc_tl                        num_il_tl
##                     0.0000000000                     0.0000000000
##                     num_op_rev_tl                    num_rev_accts
##                     0.0000000000                     0.0000000000
##              num_rev_tl_bal_gt_0                         num_sats
##                     0.0000000000                     0.0000000000
##                  num_tl_120dpd_2m                    num_tl_30dpd
##                     0.0257831206                     0.0000000000
##                num_tl_90g_dpd_24m              num_tl_op_past_12m
##                     0.0000000000                     0.0000000000
##                    pct_tl_nvr_dlq                 percent_bc_gt_75
##                     0.0000000000                     0.0124534077
##              pub_rec_bankruptcies                        tax_liens
##                     0.0000000000                     0.0000000000
##                  tot_hi_cred_lim                total_bal_ex_mort
##                     0.0000000000                     0.0000000000
##                   total_bc_limit        total_il_high_credit_limit
##                     0.0000000000                     0.0000000000
##                     hardship_flag                     hardship_dpd
##                     0.0000000000                     0.9996790995
##   hardship_last_payment_amount              disbursement_method
##                     0.9999876577                     0.0000000000
##              debt_settlement_flag                  settlement_term
##                     0.0000000000                     0.9950137000
##                            annRet                       actualTerm
##                     0.0000000000                     0.0000000000
##                      actualReturn           propSatisBankcardAccts
##                     0.0000000000                     0.0000000000
##                       borrHistory               ratio_openAccounts
##                     0.0000000000                     0.0000000000
##                  prop_delinquent
##                     0.0000000000
```

```
# or, get only those columns where there are missing values
colMeans(is.na(lcdf))[colMeans(is.na(lcdf))>0]
```

```
##                         emp_title           mths_since_last_delinq
##                     0.0630939745                     0.4782898472
##            mths_since_last_record                       revol_util
##                     0.8178642838                     0.0004196391
##                      last_pymnt_d               last_credit_pull_d
##                     0.0005554047                     0.0001481079
##       mths_since_last_major_derog                   bc_open_to_buy
##                     0.7033521760                     0.0120584533
##                           bc_util              mo_sin_old_il_acct
##                     0.0126879119                     0.0375330157
##              mths_since_recent_bc          mths_since_recent_bc_dlq
##                     0.0112068327                     0.7287526845
##             mths_since_recent_inq mths_since_recent_revol_delinq
##                     0.1014786107                     0.6293352423
```

```
##              num_tl_120dpd_2m               percent_bc_gt_75
##                  0.0257831206                   0.0124534077
##                  hardship_dpd   hardship_last_payment_amount
##                  0.9996790995                   0.9999876577
##               settlement_term
##                  0.9950137000
```

```
#remove variables which have more than 60% missing values
nm<-names(lcdf)[colMeans(is.na(lcdf))>0.6]
lcdf <- lcdf %>% select(-nm)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(nm)` instead of `nm` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
#Impute missing values - first get the columns with missing values
colMeans(is.na(lcdf))[colMeans(is.na(lcdf))>0]
```

```
##             emp_title mths_since_last_delinq             revol_util
##          0.0630939745           0.4782898472           0.0004196391
##          last_pymnt_d     last_credit_pull_d         bc_open_to_buy
##          0.0005554047           0.0001481079           0.0120584533
##               bc_util     mo_sin_old_il_acct   mths_since_recent_bc
##          0.0126879119           0.0375330157           0.0112068327
##  mths_since_recent_inq       num_tl_120dpd_2m       percent_bc_gt_75
##          0.1014786107           0.0257831206           0.0124534077
```

```
#summary of data in these columns
nm<- names(lcdf)[colSums(is.na(lcdf))>0]
summary(lcdf[, nm])
```

```
##   emp_title          mths_since_last_delinq   revol_util
##  Length:81022       Min.   :  0.00         Min.   :  0.00
##  Class :character   1st Qu.: 15.00         1st Qu.: 36.50
##  Mode  :character   Median : 30.00         Median : 54.35
##                     Mean   : 33.63         Mean   : 54.22
##                     3rd Qu.: 49.00         3rd Qu.: 72.20
##                     Max.   :133.00         Max.   :184.60
##                     NA's   :38752          NA's   :34
##   last_pymnt_d                  last_credit_pull_d bc_open_to_buy
##  Min.   :2014-09-01 00:00:00   Length:81022       Min.   :     0
##  1st Qu.:2016-01-01 00:00:00   Class :character   1st Qu.:  1025
##  Median :2017-01-01 00:00:00   Mode  :character   Median :  3656
##  Mean   :2016-11-05 04:39:00                      Mean   :  8854
##  3rd Qu.:2017-10-01 00:00:00                      3rd Qu.: 10377
##  Max.   :2018-08-01 00:00:00                      Max.   :264424
##  NA's   :45                                       NA's   :977
##     bc_util       mo_sin_old_il_acct mths_since_recent_bc mths_since_recent_inq
##  Min.   :  0.00   Min.   :  1.0      Min.   :  0.00       Min.   : 0.000
##  1st Qu.: 43.30   1st Qu.: 96.0      1st Qu.:  6.00       1st Qu.: 2.000
##  Median : 67.30   Median :128.0      Median : 13.00       Median : 5.000
##  Mean   : 63.46   Mean   :124.8      Mean   : 23.92       Mean   : 6.731
##  3rd Qu.: 87.40   3rd Qu.:152.0      3rd Qu.: 29.00       3rd Qu.:10.000
##  Max.   :318.20   Max.   :545.0      Max.   :451.00       Max.   :25.000
##  NA's   :1028     NA's   :3041       NA's   :908          NA's   :8222
```

```
##  num_tl_120dpd_2m percent_bc_gt_75
##  Min.   :0.000    Min.   :  0.00  
##  1st Qu.:0.000    1st Qu.: 20.00  
##  Median :0.000    Median : 50.00  
##  Mean   :0.001    Mean   : 49.43  
##  3rd Qu.:0.000    3rd Qu.: 80.00  
##  Max.   :3.000    Max.   :100.00  
##  NA's   :2089     NA's   :1009    
```

```
#mths_since_last_delinq: has 48% missings, these pertain to no delinquincy, so replace by max value (17
lcx<-lcdf[, c(nm)]
colMeans(is.na(lcx))[colMeans(is.na(lcx))>0]
```

```
##              emp_title mths_since_last_delinq             revol_util 
##           0.0630939745           0.4782898472           0.0004196391 
##           last_pymnt_d     last_credit_pull_d         bc_open_to_buy 
##           0.0005554047           0.0001481079           0.0120584533 
##                bc_util     mo_sin_old_il_acct   mths_since_recent_bc 
##           0.0126879119           0.0375330157           0.0112068327 
##  mths_since_recent_inq       num_tl_120dpd_2m       percent_bc_gt_75 
##           0.1014786107           0.0257831206           0.0124534077
```

```
lcx<- lcx %>% replace_na(list(mths_since_last_delinq = 500))
#For revol_util, suppose we want to replace the misisng values by the median
lcx<- lcx %>% replace_na(list(revol_util=median(lcx$revol_util, na.rm=TRUE)))

#Similarly for the other variables
#After trying this out on the temporary dataframe lcx, if we are sure this is what we want, we can now
lcdf<- lcdf %>% replace_na(list(mths_since_last_delinq=500, revol_util=median(lcdf$revol_util, na.rm=TRU


#Have we addressed all missing values ?
colMeans(is.na(lcdf))[colMeans(is.na(lcdf))>0]
```

```
##          emp_title       last_pymnt_d last_credit_pull_d 
##       0.0630939745       0.0005554047       0.0001481079
```

```
  #You will see that last_pymnt_d still have a few missing values - do you understand what these missin
  # Are they probably for the charged-off loans ?  You can check:
  #lcdf %>% filter(is.na(lcdf$last_pymnt_d)) %>% group_by(loan_status) %>% tally()
```

Question 3

Consider the potential for data leakage. You do not want to include variables in your model which may not be available when applying the model; that is, some data may not be available for new loans before they are funded. Leakage may also arise from variables in the data which may have been updated during the loan period (ie., after the loan is funded). Identify and explain which variables will you exclude from the model.

```
lcdf <- lcdf %>% select(-c(funded_amnt_inv, term, emp_title, pymnt_plan, title, zip_code, addr_state, o

#Drop some other variables
varsToRemove <- c("actualTerm","annRet")
lcdf <- lcdf %>% select(-varsToRemove)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(varsToRemove)` instead of `varsToRemove` to silence this message.
```

```
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

Question 4

Do a uni-variate analyses to determine which variables (from among those you decide to consider for the next stage prediction task) will be individually useful for predicting the dependent variable (loan_status). For this, you need a measure of relationship between the dependent variable and each of the potential predictor variables. Given loan-status as a binary dependent variable, which measure will you use? From your analyses using this measure, which variables do you think will be useful for predicting loan_status ?

(Note – if certain variables on their own are highly predictive of the outcome, it is good to ask if this variable has a leakage issue).

Uni-variate analyses - which variables are individually predictive of the outcome ?

```
#Can compute the AUC for each variable

lcdf <- lcdf %>% mutate_if(is.character, as.factor)
library(pROC) #this package has a function auc(..) which we can readily use

#We will use the function auc(response, prediction) which returns the AUC value for the specified predi
auc(response=lcdfTrn$loan_status, lcdfTrn$loan_amnt)
```

```
## Area under the curve: 0.5123
```

```
 # returns the value for loan_amt as predictor

#In the auc(..) function, the predictor variable has to be numeric  - otherwise, how would it calculate
auc(response=lcdfTrn$loan_status, as.numeric(lcdfTrn$emp_length))
```

```
## Area under the curve: 0.5282
```

```
# There are a few date type variables - we will ignore these here.

#How would you do this for all variables in the dataset?
# Rather than call the function individually for each variable, we can use the sapply(..) function

# For the numeric variables:
aucsNum<-sapply(lcdfTrn %>% select_if(is.numeric), auc, response=lcdfTrn$loan_status)

#Or considering both numeric and factor variables:
aucAll<- sapply(lcdfTrn %>% mutate_if(is.factor, as.numeric) %>% select_if(is.numeric), auc, response=l

#TO determine which variables have auc > 0.5
aucAll[aucAll>0.5]
```

```
##             loan_amnt            emp_length                   dti
##             0.5122576             0.5282331             0.5738898
##          actualReturn propSatisBankcardAccts           borrHistory
##             0.9855016             0.5223215             0.5441877
##     ratio_openAccounts
##             0.5403054
```

```r
#Or, we can use the tidy(..) function from the broom package - which converts the 'messy' output into a
library(broom)

tidy(aucAll[aucAll > 0.5]) %>% view()

# or  in any range of values like, tidy(aucAll[aucAll >=0.5 & aucAll < 0.6])
# or in sorted order
tidy(aucAll) %>% arrange(desc(aucAll))
```

```
## # A tibble: 10 x 2
##    names                      x
##    <chr>                  <dbl>
##  1 actualReturn           0.986
##  2 dti                    0.574
##  3 borrHistory            0.544
##  4 ratio_openAccounts     0.540
##  5 emp_length             0.528
##  6 propSatisBankcardAccts 0.522
##  7 loan_amnt              0.512
##  8 prop_delinquent        0.499
##  9 purpose                0.475
## 10 total_rec_late_fee     0.453
```

Based on the tables using the auc score we were able to make certain assumptions. When considering what will be the most useful for predicting loan_status, there are 3 keys variables that can help in this. sub_grade, grade, and int_rate are very telling about the status. Loan grades are associated with the danger in giving a loan. an A would mean there is lower risk in defaulting than other loans like a D. This is shown with an auc score of .6703419. Another strong variable that helps with determining loan_status is annual_inc. By itself income will tell you that a borrower has more money to potentially pay back, but linked with a variable like grade or interest rate it can give more information about the type of loan or where payment can stand. There is no perfection to predicting loan_status but with these variables.