UDACITY

‹ Back to Data Analyst Nanodegree

Explore and Summarize Data

REVIEW

CODE REVIEW

HISTORY

Requires Changes

3 SPECIFICATIONS REQUIRE CHANGES

Code Functionality

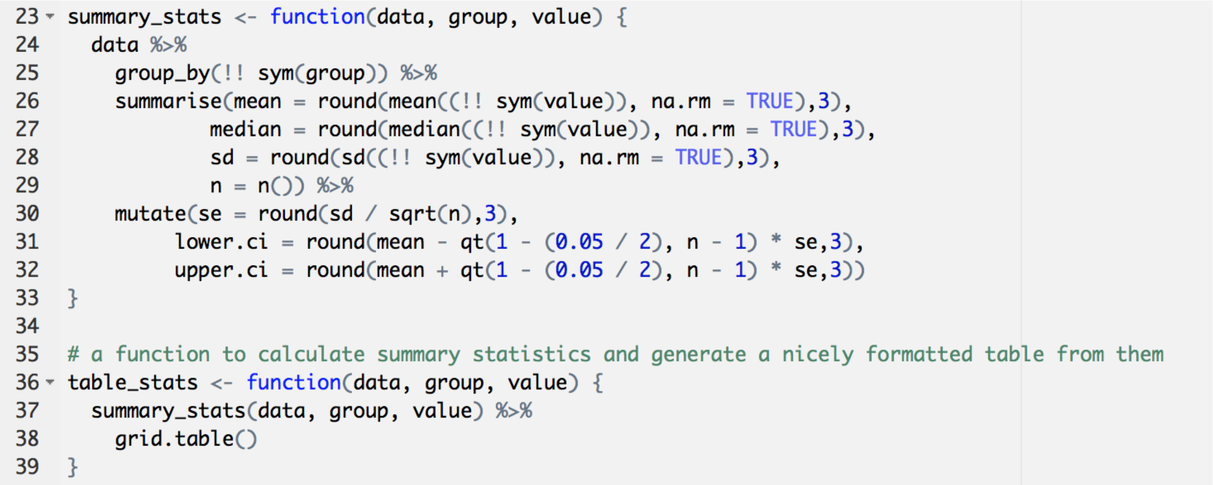
✓

All code is functional (e.g. No Error is produced and RMD document is not prevented from being knit.)

↻

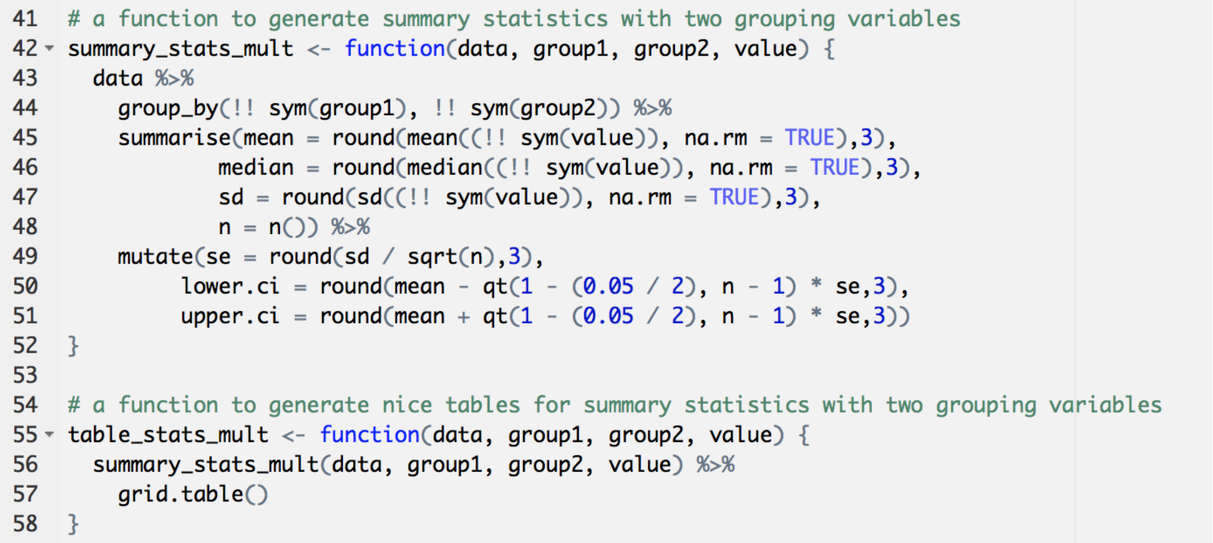
The project almost never uses repetitive code where a function would be more appropriate. The code references variables by name instead of using constants or column numbers.

1. There are two function definitions that are repetitive: `table_stats` and `table_stats_mult`, since they share majority of the code that's also used in `summary_stats` and `summary_stats_mult` correspondingly, you could use those two functions and add `grid.table()` to make new functions `table_stats` and `table_stats_mult`, like shown below:



```r
summary_stats <- function(data, group, value) {
  data %>%
    group_by(!! sym(group)) %>%
    summarise(mean = round(mean((!! sym(value)), na.rm = TRUE),3),
            median = round(median((!! sym(value)), na.rm = TRUE),3),
            sd = round(sd((!! sym(value)), na.rm = TRUE),3),
            n = n()) %>%
    mutate(se = round(sd / sqrt(n),3),
         lower.ci = round(mean - qt(1 - (0.05 / 2), n - 1) * se,3),
         upper.ci = round(mean + qt(1 - (0.05 / 2), n - 1) * se,3))
}

# a function to calculate summary statistics and generate a nicely formatted table from them
table_stats <- function(data, group, value) {
  summary_stats(data, group, value) %>%
    grid.table()
}
```

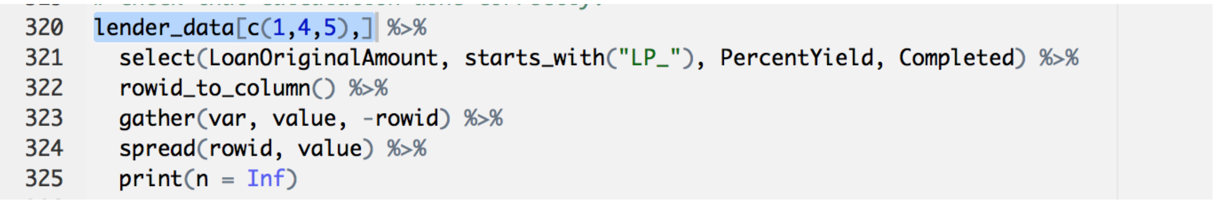


```r
# a function to generate summary statistics with two grouping variables
summary_stats_mult <- function(data, group1, group2, value) {
  data %>%
    group_by(!! sym(group1), !! sym(group2)) %>%
    summarise(mean = round(mean((!! sym(value)), na.rm = TRUE),3),
            median = round(median((!! sym(value)), na.rm = TRUE),3),
            sd = round(sd((!! sym(value)), na.rm = TRUE),3),
            n = n()) %>%
    mutate(se = round(sd / sqrt(n),3),
         lower.ci = round(mean - qt(1 - (0.05 / 2), n - 1) * se,3),
         upper.ci = round(mean + qt(1 - (0.05 / 2), n - 1) * se,3))
}

# a function to generate nice tables for summary statistics with two grouping variables
table_stats_mult <- function(data, group1, group2, value) {
  summary_stats_mult(data, group1, group2, value) %>%
    grid.table()
}
```

which will make the code not repeating that much, nicer to read

2. You used column numbers instead of names in following code:



```r
lender_data[c(1,4,5),] %>%
  select(LoanOriginalAmount, starts_with("LP_"), PercentYield, Completed) %>%
  rowid_to_column() %>%
  gather(var, value, -rowid) %>%
  spread(rowid, value) %>%
  print(n = Inf)
```

Please try to use column names to reference columns, since it's more resistant to data corruption(missing columns or etc.), and makes the code more readable(so that you know which columns are you referencing to)

Project Readability

✓

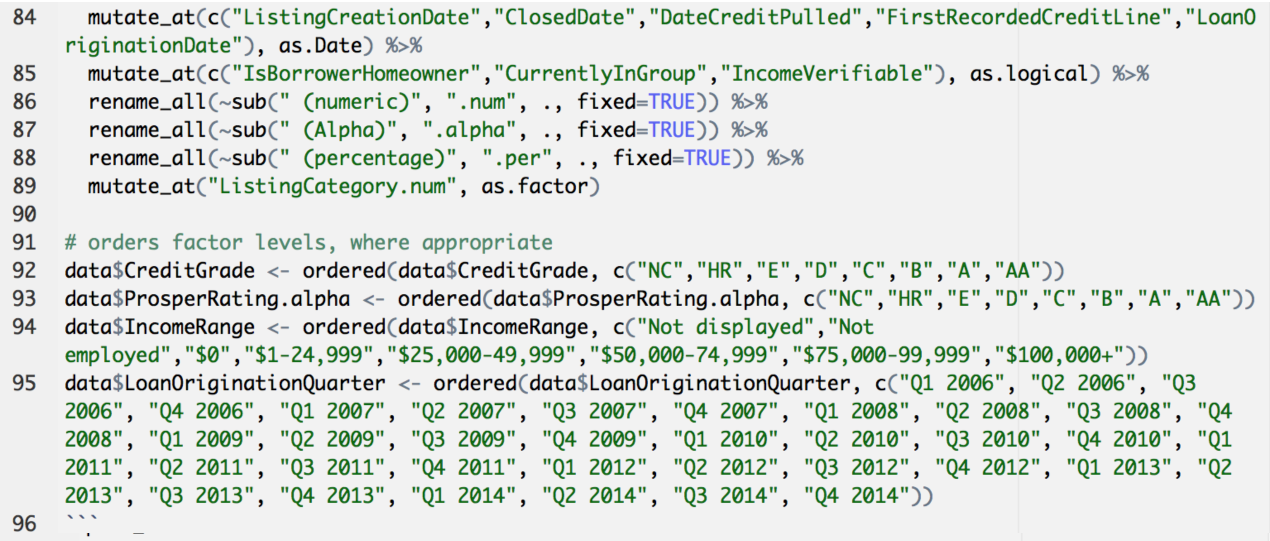
All complex code is adequately explained with comments. It is always clear what the code is doing and how and why any unusual coding decisions were made.

For a complicated project like this, your comments really help reviewers see what you're doing through code comments, good job 👍

🔄

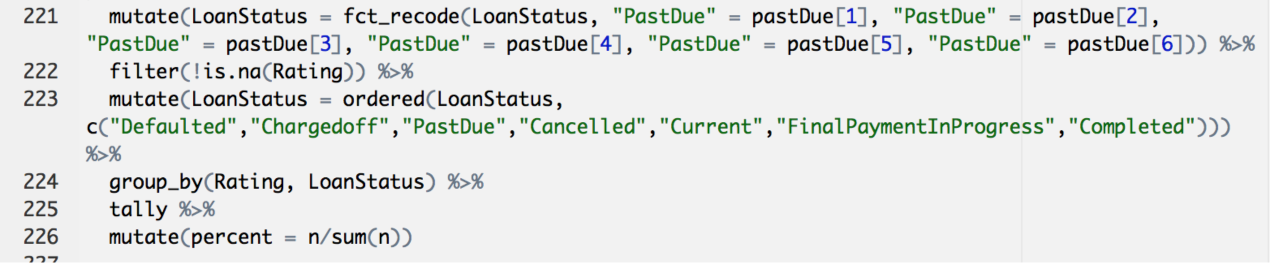
The code uses formatting techniques in a consistent and effective manner to improve code readability. All lines are shorter than 80 characters.

There are many lines exceed 80 character length limit, two examples are shown below:



```
  mutate_at(c("ListingCreationDate","ClosedDate","DateCreditPulled","FirstRecordedCreditLine","LoanOriginationDate"), as.Date) %>%
  mutate_at(c("IsBorrowerHomeowner","CurrentlyInGroup","IncomeVerifiable"), as.logical) %>%
  rename_all(~sub(" (numeric)", ".num", ., fixed=TRUE)) %>%
  rename_all(~sub(" (Alpha)", ".alpha", ., fixed=TRUE)) %>%
  rename_all(~sub(" (percentage)", ".per", ., fixed=TRUE)) %>%
  mutate_at("ListingCategory.num", as.factor)

# orders factor levels, where appropriate
data$CreditGrade <- ordered(data$CreditGrade, c("NC","HR","E","D","C","B","A","AA"))
data$ProsperRating.alpha <- ordered(data$ProsperRating.alpha, c("NC","HR","E","D","C","B","A","AA"))
data$IncomeRange <- ordered(data$IncomeRange, c("Not displayed","Not employed","$0","$1-24,999","$25,000-49,999","$50,000-74,999","$75,000-99,999","$100,000+"))
data$LoanOriginationQuarter <- ordered(data$LoanOriginationQuarter, c("Q1 2006", "Q2 2006", "Q3 2006", "Q4 2006", "Q1 2007", "Q2 2007", "Q3 2007", "Q4 2007", "Q1 2008", "Q2 2008", "Q3 2008", "Q4 2008", "Q1 2009", "Q2 2009", "Q3 2009", "Q4 2009", "Q1 2010", "Q2 2010", "Q3 2010", "Q4 2010", "Q1 2011", "Q2 2011", "Q3 2011", "Q4 2011", "Q1 2012", "Q2 2012", "Q3 2012", "Q4 2012", "Q1 2013", "Q2 2013", "Q3 2013", "Q4 2013", "Q1 2014", "Q2 2014", "Q3 2014", "Q4 2014"))
```



```
  mutate(LoanStatus = fct_recode(LoanStatus, "PastDue" = pastDue[1], "PastDue" = pastDue[2], "PastDue" = pastDue[3], "PastDue" = pastDue[4], "PastDue" = pastDue[5], "PastDue" = pastDue[6])) %>%
  filter(!is.na(Rating)) %>%
  mutate(LoanStatus = ordered(LoanStatus, c("Defaulted","Chargedoff","PastDue","Cancelled","Current","FinalPaymentInProgress","Completed")))
%>%
  group_by(Rating, LoanStatus) %>%
  tally %>%
  mutate(percent = n/sum(n))
```

RStudio can draw a vertical line at 80 characters for you so that it's easy to check. This webpage shows how to do that with 'show margin':
https://support.rstudio.com/hc/en-us/articles/200549016-Customizing-RStudio#editing

Please note that above two plots are not the only violating lines of code, please use the help of the `show margin` line to find out all of the lines and fix them in your next submission

✓

Markdown syntax is used in the RMD file to improve readability of the knitted file.

Perfect formatting. I can see you used css style file and some javascript to control format, feel like a little bit overkill but still good job 👍

Quality of Analysis

✓

The project appropriately uses univariate, bivariate, and multivariate plots to explore most of the expected relationships in the data set.

✓

Questions and findings are placed between blocks of R code regularly so it is clear what the student was thinking throughout the analysis.

Very detailed findings/comments throughout the project, I can see clearly how you did analysis and jumping from one plot to next, the logic flow is very clear to me 👍

✓

Reasoning is provided for the plots made throughout the analysis. Plots made follow a logical flow. Comments following plots accurately reflect the plots' contents.

same as above

✓

The project contains at least 20 visualizations. The visualizations are varied and show multiple comparisons and trends. Relevant statistics (e.g. mean, median, confidence intervals, correlations) are computed throughout the analysis when an inference is made about the data.

↻

Visualizations made in the project depict the data in an appropriate manner that allows plots to be readily interpreted. Choice of plot type, variables, and aesthetic parameters (e.g. bin width, color, axis breaks) is appropriate.

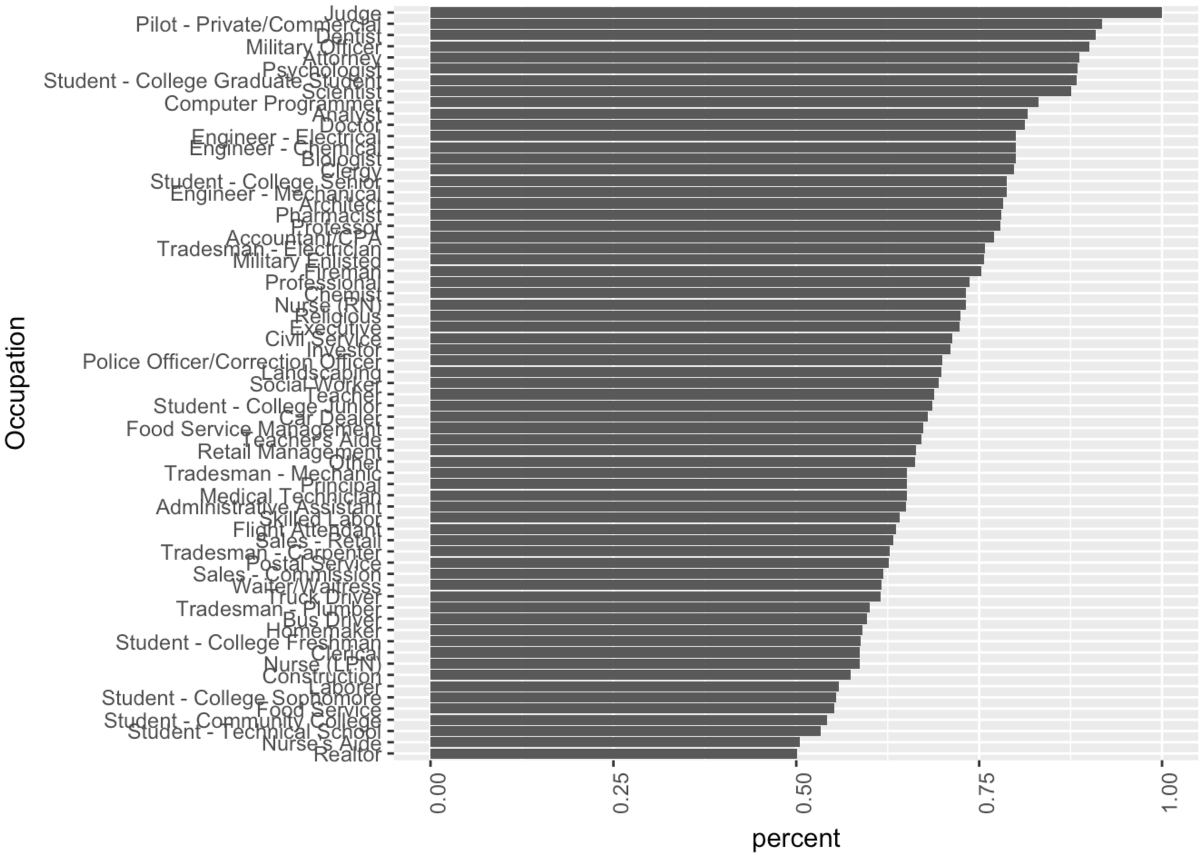
First of all good job on choosing proper plot types for most of the plots! You definitely put a lot of effort into this project, which will be a valuable experience for you to back up your future projects!
A few suggestions as follows, both required and optional:

Required changes

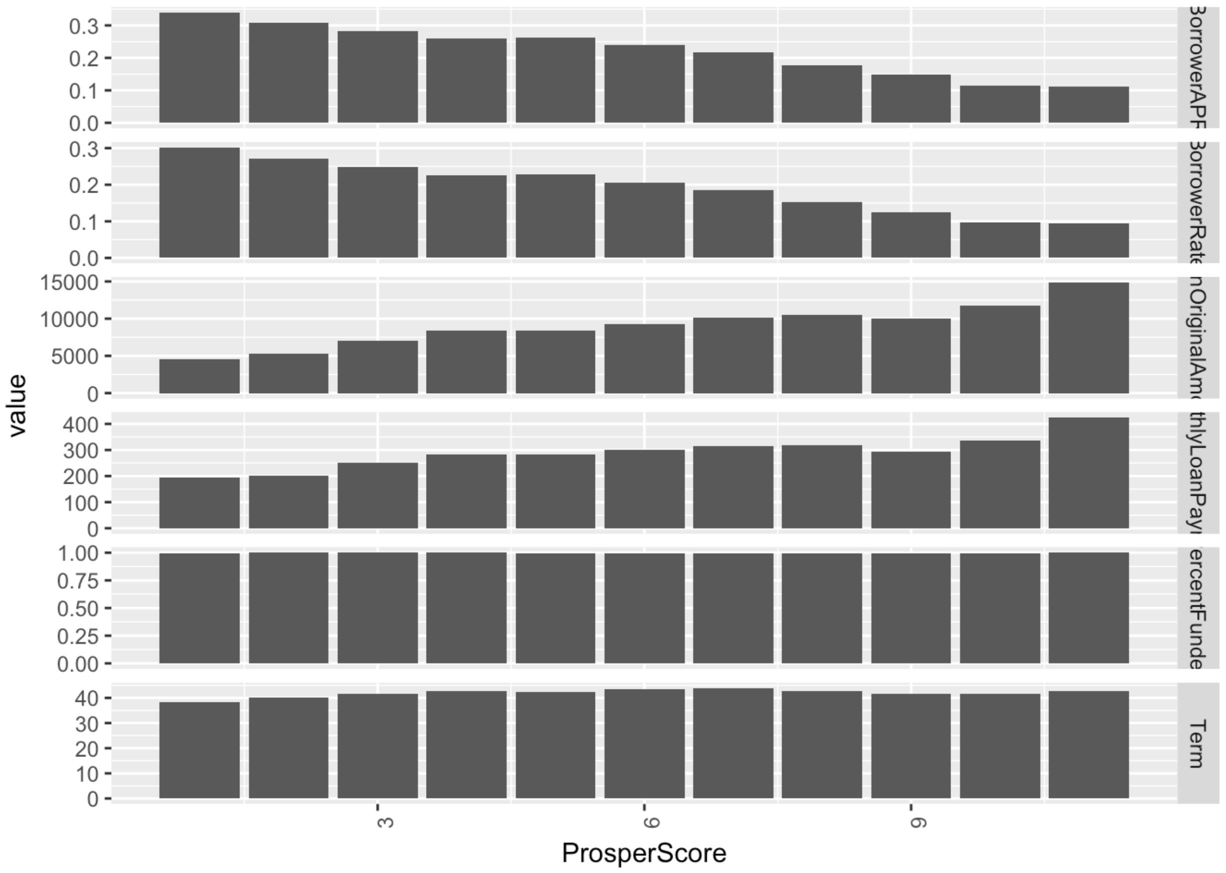
Make plots big enough to show all details

Following 3 plots are kind of small such that they cut off either variable names, or have overlapping axis labels:

`Completed`

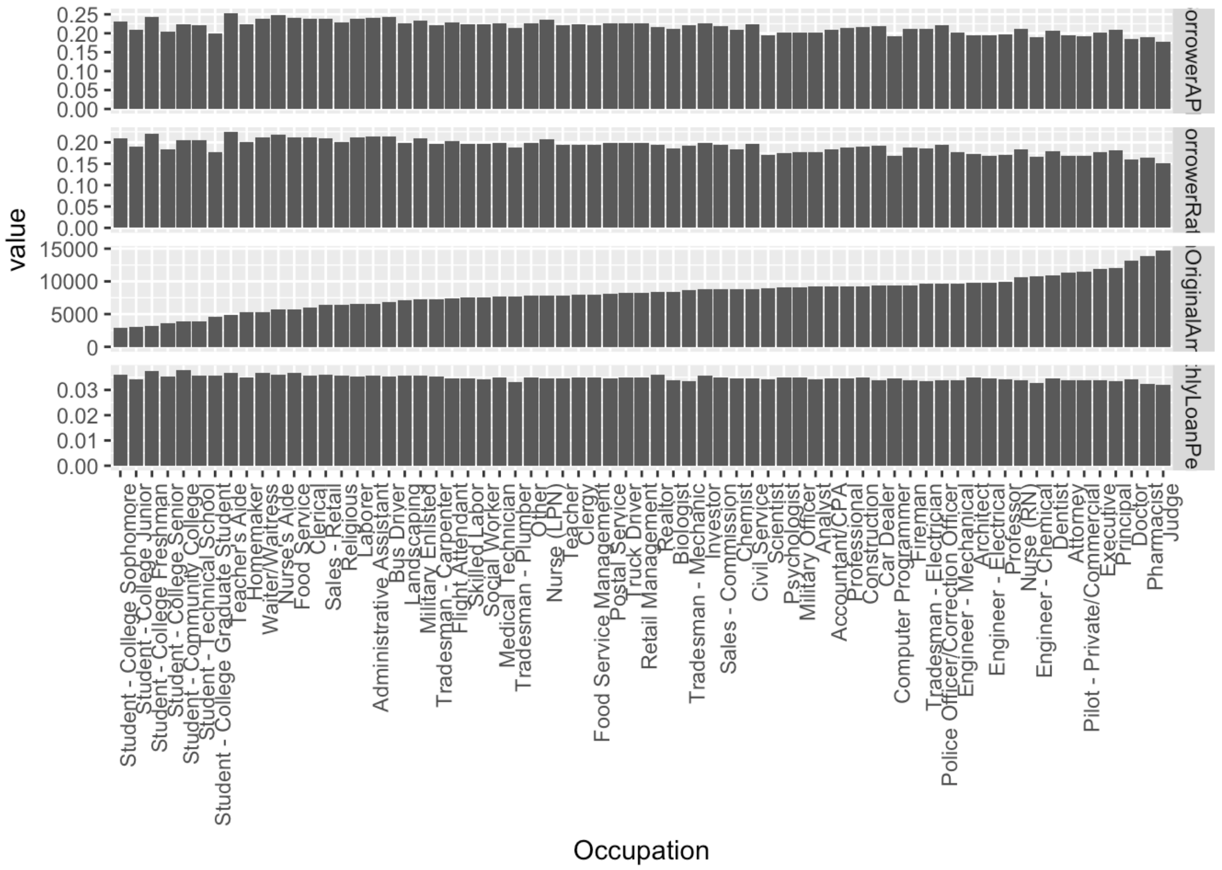


`ProsperScore`



`Occupation`

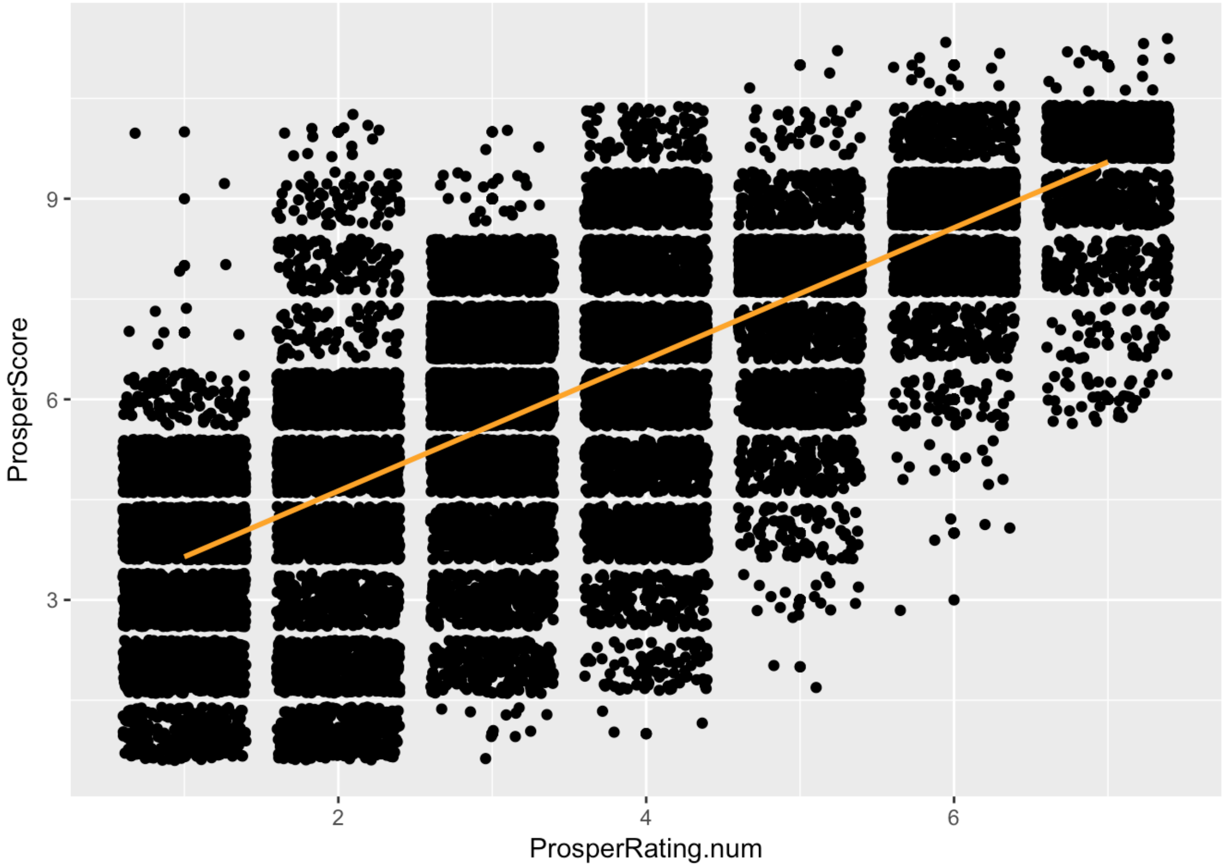
Again, I expect higher-paying occupations will receive better terms, on average.



Please make plots sizes larger, by changing values of `fig.height` and `fig.width` to proper values

Overlapping data points should be more transparent

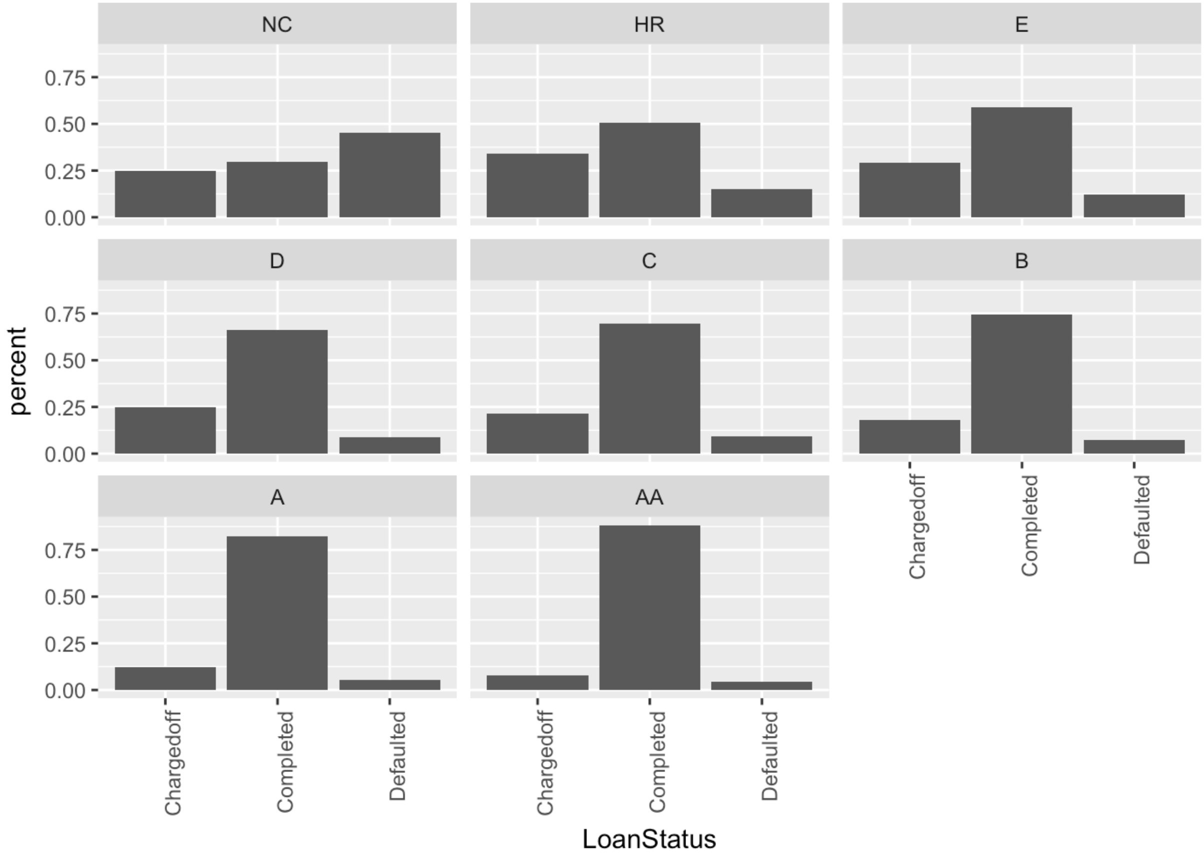
For following plot, please add alpha value for data points, so it's easier to see distribution of those data points:



Optional change

For following plot, it's better to read if you rotate x axis labels by 45 degrees clockwise

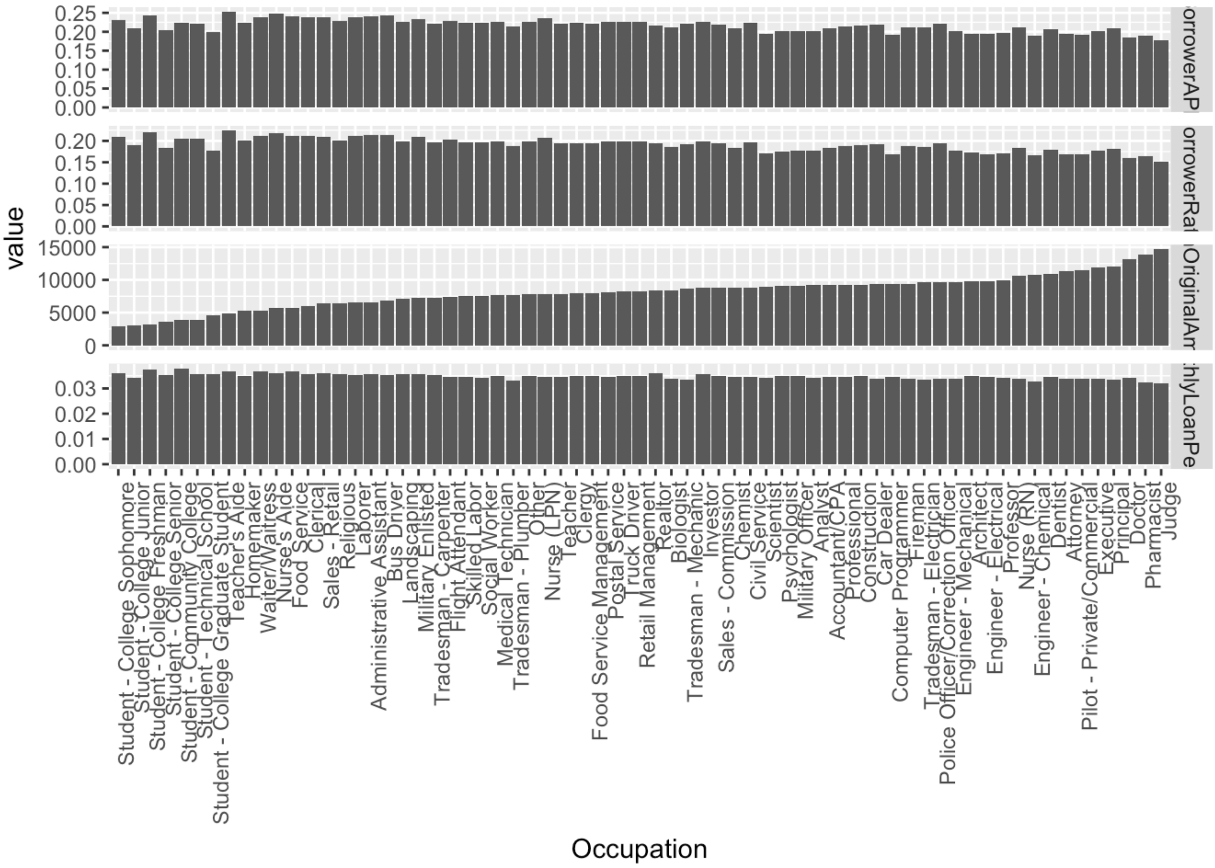
`LoanStatus`



Similarly for following plot, it's easier to read if you rotate the plot 90 degrees(use `coord_flip()` or similar functions you can find)

`Occupation`

Again, I expect higher-paying occupations will receive better terms, on average.



Final Plots and Summary

✓

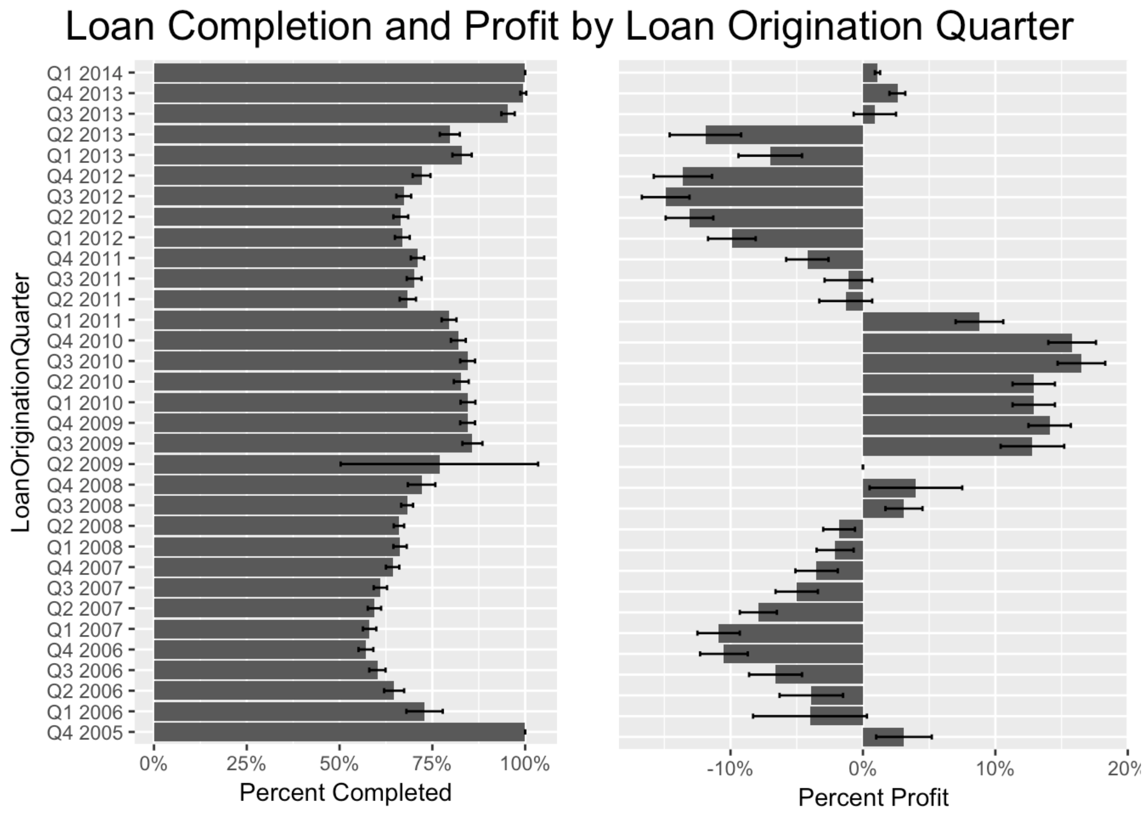
The project includes a Final Plots and Summary section containing three plots and commentary. All plots in this section reflect what has been explored in the main body of the analysis.

✓

The plots are well chosen and the plots fulfill at least 2 of the criteria. The plots are varied and reveal interesting trends and relationships.

Very interesting trends found, awesome job! Also for Final Plot 3:

Lender profit by loan origination quarter



I also noticed a seasonal trend that happens during every 2 years span. Maybe you can elaborate on that one later too, if you're interested

✓

All plots have appropriately selected variables and are plotted in a way that accurately conveys the data/information (i.e findings in Final Plot 1 do not depend on the findings of Final Plot 2).

✓

All plots are labeled appropriately (axis labels, plot titles, axis units) and can be read and interpreted easily. Plots are scaled appropriately.

✓

The reasoning and findings from each plot are explained and the text about each plot is descriptive enough to stand alone. Comments reflect the contents of the plots that they are associated with.

Reflection

✓

The project includes a Reflection section discussing the analysis performed.

✓

The section reflects on how the analysis was conducted and reports on the struggles and successes throughout the analysis. The section provides at least one idea or question for future work. The section explains any important decisions in the analysis and how those decisions affected the analysis.

For this particular dataset, you've already provided your own learning experience, and ideas for future work, which is awesome 😃
Please also keep in mind this dataset has very skewed variables and some outliers, you can notice that when doing boxplots for some variables, where boxplot shrink to dots and straight lines, which gives us the message that the variable has outliers, or has very skewed distribution for those variables, you can add those possibilities to future work too, especially when you're already thinking about building predictive models, but not considering skewness or outliers in the dataset yet, those two can be two of your biggest enemies when you try to build powerful models. Doing data transformation and trimming outliers can help you with model building process, please keep that in mind for your future projects.

☑ RESUBMIT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Student FAQ