DS_5100_HW10

Eashan Kaw

4/9/2023

Metadata

```
Course:  DS 5100
Module:  10 R Programming 1
Topic:   HW Computing Payoff for a Quota Structure
Author:  R.C. Alvarado (adapted)
Date:    06 October 2022 (revised)
```

Student Info

```
Name: Eashan Kaw
Ned ID: elk7ed
URL of this file on GitHub: https://github.com/elkaw/DS5100-2023-01-O/blob/main/lessons/M10_RBasics/M10-HW.ipynb
```

Instructions

In your **private course repo** use this notebook to write code that performs the tasks below.

Save your notebook in the `M10` directory.

Remember to add and commit these files to your repo.

Then push your commits to your repo on GitHib.

Be sure to fill out the **Student Info** block above.

To submit your homework, save your results as a PDF and upload it to GradeScope.

TOTAL POINTS: 12

Overview

A salesperson at a large tech firm is faced with a new payment structure.

This salesperson has a quarterly quota of $225,000.

The payment received follows a progressive schedule with four brackets as follows:

1. For the first 40% of quota, the salesperson receives 7% on quota reached
2. For the next 30% of quota, the salesperson receives 10% on quota reached
3. For the next 20% of quota, the salesperson receives 13% on quota reached
4. For the next 10% of quota, the salesperson receives 16% on quota reached

For example, if the salesperson is 50% to quota, reaching $112,500 of sales, then:

- `a` = the first 40% is paid out at 7%, thus payout = $225,000 * 40% * 7%
- `b` = the next 10% is paid out at 10%, thus payout = $225,000 * 10% * 10%

The total payout to the salesperson would be `a + b`.

Notice what does *not* happen: getting to the second bracket does NOT mean the payout is $225,000 * 50% * 10%.

In another example, a salesperson is at 20% quota. Their payout would be $225,000 * 20% * 7%.

This schedule represents earnings up to 100% of quota. We ignore sales above 100% here.

Given this, the salesperson would like to know how much she would earn if she reaches a given percentage of quarterly quota.

Note: The quota structure in this assignment is analogous to how the US tax system works: There are several **brackets** with rate r applied to dollars in bracket i.

Task 1

(4 points)

Create a dataframe that encodes the information presented in the question. That is, asswume that each row of the dataframe stands for a bracket, and that the columns stand for the features described in the progressive schedule. Then, using the quarterly quota of $225,000, add columns to the dataframe that apply the encoded parameters to this value for each bracket. You should end up with columns for the earnings in dollars for each

bracket, as well as the payout in dollars.

```
# CODE HERE
library("formattable")
```

```
## Warning: package 'formattable' was built under R version 4.0.2
```

```
library("dplyr")
```

```
## Registered S3 methods overwritten by 'tibble':
##   method     from
##   format.tbl pillar
##   print.tbl  pillar
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
parameter_df <- data.frame (
  bracket_rate = c(.4, .3, .2, .1),
  commission_rate  = c(.07, .10, .130, .16)
)

parameter_df$quota_reached <- cumsum(parameter_df$bracket_rate)
parameter_df$earnings <-225000 * parameter_df$bracket_rate
parameter_df$payout <- parameter_df$earnings * parameter_df$commission_rate
parameter_df$cum_earnings <- cumsum(parameter_df$earnings )
parameter_df$cum_payout <- cumsum(parameter_df$payout)
parameter_df_formatted <- parameter_df
parameter_df_formatted$earnings <- sprintf("$%.2f",parameter_df_formatted$earnings) #look for commands
parameter_df_formatted$payout <- sprintf("$%.2f",parameter_df_formatted$payout)
parameter_df_formatted$bracket_rate <- formattable::percent(parameter_df_formatted$bracket_rate )
parameter_df_formatted$commission_rate <- formattable::percent(parameter_df_formatted$commission_rate )
parameter_df_formatted$quota_reached <- formattable::percent(parameter_df_formatted$quota_reached ) # percent fun
ction of formattable
parameter_df_formatted$cum_earnings <- sprintf("$%.2f",parameter_df_formatted$cum_earnings )
parameter_df_formatted$cum_payout <- sprintf("$%.2f",parameter_df_formatted$cum_payout )
parameter_df
```

```
##   bracket_rate commission_rate quota_reached earnings payout cum_earnings
## 1          0.4            0.07           0.4    90000   6300        90000
## 2          0.3            0.10           0.7    67500   6750       157500
## 3          0.2            0.13           0.9    45000   5850       202500
## 4          0.1            0.16           1.0    22500   3600       225000
##   cum_payout
## 1       6300
## 2      13050
## 3      18900
## 4      22500
```

Task 2

(4 points)

Write a function that takes an argument for the fraction of quarterly quota reached by the salesperson, expressed as a decimal value between 0 and 1 (e.g. 0.8 means 80%), and which returns the dollar amount earned.

This function should use the previously defined dataframe as a global variable. Note that this function is greatly simplified if your first dataframe has cumulative sums for the dollar amount columns.

Do not use for loops in completing this task or the next. Instead, let your dataframe do the work. In your function, match the amount earned to the appropriate row in your first dataframe to get the answer. In applying your function, use `apply()` and assign the result as a second column to your second dataframe.

```
# CODE HERE
#typeof()
#as.numeric(gsub('[$,]', '', parameter_df$earnings)) #source https://stackoverflow.com/questions/31944103/convert
-currency-with-commas-into-numeric

calculate_payout <- function(quota_fraction){
  payout <- 0
  bracket <- min(parameter_df$cum_earnings[parameter_df$cum_earnings >= quota_fraction*225000] , na.rm=T)
  if( length(parameter_df$cum_earnings[parameter_df$cum_earnings < quota_fraction*225000]) == 0){
    payout <- (quota_fraction*225000)*parameter_df$commission_rate[parameter_df$cum_earnings == bracket]
    return(payout)
  }
  bracket_before <- max(parameter_df$cum_earnings[parameter_df$cum_earnings < quota_fraction*225000] , na.rm=F)
  payout <- parameter_df$cum_payout[parameter_df$cum_earnings == bracket_before] + (quota_fraction*225000 - brack
et_before)*parameter_df$commission_rate[parameter_df$cum_earnings == bracket]
  return(payout)

}
print(calculate_payout(.5))
```

```
## [1] 8550
```

```
payout_df <- data.frame (quota_earned = c(.5))
payout_df$payout <- apply(payout_df,1,calculate_payout)
payout_df
```

```
##   quota_earned payout
## 1          0.5   8550
```

Task 3

(2 points)

Call the function to get the dollar amount earned in increments of 10% in a range between 0% to 100% earned. Note that you can use `seq()` to generate these increments.

Be sure to put the results of your function at work into a second dataframe. That is, create a dataframe with columns for percent of quota earned and payout for that amount.

```
# CODE HERE
payout_df <- data.frame(quota_earned = seq(0,1,.1))
payout_df$payout <- apply(payout_df,1,calculate_payout)
payout_df
```

```
##    quota_earned payout
## 1           0.0      0
## 2           0.1   1575
## 3           0.2   3150
## 4           0.3   4725
## 5           0.4   6300
## 6           0.5   8550
## 7           0.6  10800
## 8           0.7  13050
## 9           0.8  15975
## 10          0.9  18900
## 11          1.0  22500
```

Task 4

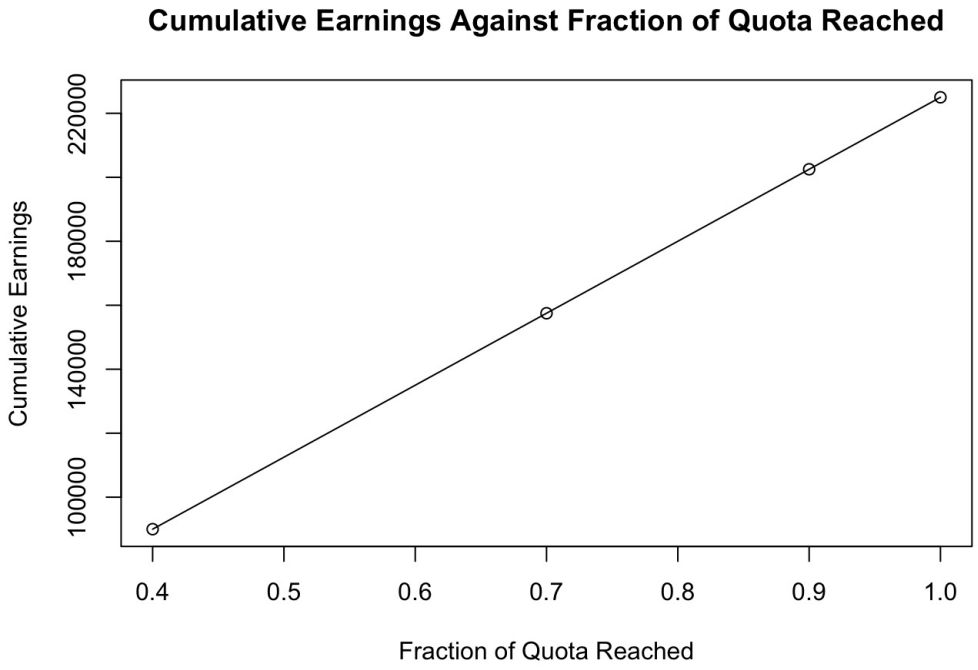
(1 point)

Using the first dataframe, plot the amounts earned (y-axis) versus quarterly quota reached (x-axis).

Display the graph using both points and lines.

Hint: for both axes, use the cumulative sums, which you should have defined above.

```
plot(parameter_df$quota_reached, parameter_df$cum_earnings, , xlab='Fraction of Quota Reached' , ylab='Cumulative
Earnings', main = 'Cumulative Earnings Against Fraction of Quota Reached')
lines(parameter_df$quota_reached[order(parameter_df$quota_reached)], parameter_df$cum_earnings[order(parameter_df
$quota_reached)], xlim=range(parameter_df$quota_reached), ylim=range(parameter_df$cum_earnings), pch=16)
```



Task 5

(1 point)

Using the second dataframe, plot the dollar amount for each increment (x-axis) versus the payout in dollars (y-axis).

Again, display the graph using both points and lines.

```
# CODE HERE
plot(payout_df$quota_earned, payout_df$payout, xlab="Fraction of Quota Earned", ylab="Payout ($)", main = "Commis
sions Payout Against Quota Earned",pch=16)
lines(payout_df$quota_earned[order(payout_df$quota_earned)], payout_df$payout[order(payout_df$quota_earned)], xli
m=range(payout_df$quota_earned), ylim=range(payout_df$payout), pch=16)
```