R Notebook

Metadata

```
Course:  DS 5100
Module:  10 R Programming 1
Topic:   HW Computing Payoff for a Quota Structure
Author:  Sae'von Palmer
Date:    29 March 2025
```

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 sales quota Q of $\$225,000$.

The commission received C follows a progressive schedule with four brackets as follows:

- **Bracket 1**. For the first 40% of Q, the salesperson receives 7% on Q reached.
- **Bracket 2**. For the next 30% of Q, the salesperson receives 10% on Q reached.
- **Bracket 3**. For the next 20% of Q, the salesperson receives 13% on Q reached.
- **Bracket 4**. For the final 10% of Q, the salesperson receives 16% on Q reached.

As an example of how to apply this schedule, consider the case of a salesperson who brings in 50% of Q, or $\$112,500$ of sales.
In this situation, the sales land in the second bracket. So we can compute C as follows:

- a = the first 40% paid out at 7%, or $\$225,000 \times 40\% \times 7\% = \$6,300$.
- b = the next 10% paid out at 10%, or $\$225,000 \times 10\% \times 10\% = \$2,250$.

The total payout C to the salesperson would be $a + b = \$8,550$.

Notice that getting to the second bracket does *not* mean the payout is $\$225,000 \times 50\% \times 10\%$.

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

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

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

Task 1

(2 points)

Assign the value of Q to `quota` .

Create a data frame called `df` that encodes the information presented in the question.

That is, assume that each row of the data frame stands for a bracket, and that the columns stand for the features described in the progressive schedule.

Initially, the data frame should have two columns:

- `cut` for the percentage value associated with the bracket, i.e. $.4, .3, \ldots$
- `payout_pct` for the percent of Q received for the bracket, i.e. $.07, .1, \ldots$

In both cases, express the percentages as a decimal value between 0 and 1.

Display `df` in a cell.

```
# assign the value of Q to quota
quota <- 225000

# Create the initial data frame
df <- data.frame(
  cut = c(0.4, 0.3, 0.2, 0.1),
  payout_per = c(0.07, 0.10, 0.13, 0.16)
)

# Display df
print(df)
```

```
##   cut payout_per
## 1 0.4       0.07
## 2 0.3       0.10
## 3 0.2       0.13
## 4 0.1       0.16
```

Task 2

(2 points)

Augment `df` with derived columns that will be used to compute C for a given amount of sales:

- `cut_sum` : The cumulative sum of `cut` , e.g. $.4$, $.7, \ldots$
- `amt` : The earned quota, i.e. product of `cut` and Q.
- `payout` : The product of `amt` and `payout_cut` .

- `payout_sum` : The cumulative sum of `payout` .
- `amt_sum` : The cumulative sum of the earned quota `amt` .

Display the augmented `df` in a cell.

```
# cumulative sum of cut
df$cut_sum <- cumsum(df$cut)

# earned quota amount
df$amt <- df$cut * quota

# payout for each bracket
df$payout <- df$amt * df$payout_per

# cumulative sum of payouts
df$payout_sum <- cumsum(df$payout)

# cumulative sum of the earned quota
df$amt_sum <- cumsum(df$amt)

# Display the updated df
print(df)
```

```
##   cut payout_per cut_sum   amt payout payout_sum amt_sum
## 1 0.4       0.07     0.4 90000   6300       6300   90000
## 2 0.3       0.10     0.7 67500   6750      13050  157500
## 3 0.2       0.13     0.9 45000   5850      18900  202500
## 4 0.1       0.16     1.0 22500   3600      22500  225000
```

Task 3

(2 points)

Write a function `get_bracket()` that will return the bracket number $k \in [1, 2, 3, 4]$ for a given amount of sales S, represented as a fraction of Q expressed as a decimal number rounded to one place.

For example, if a salesperson makes 50% of quota, then $S = .5$.

The function should use `df` and assume it is global.

Hints:

This function requires that you match S to a value in the appropriate column of `df` . To do this, you'll need to use a boolean condition comparing S to the column, and then select the index of the column that meets the condition. You can select the index using `which()` .

However, sometimes `which()` will return a vector with more than one value, so you will need to select either the `min()` or `max()` of the returned vector, depending on the condition.

Finally, to make sure that borderline values, such as $.4$, are matched with the correct bracket (e.g. $.4$ belongs to bracket 1 not 2), you should round both values in the boolean expression using `round()` .

```
get_bracket <- function(S) {
  # Ensure S is rounded to 1 decimal place
  S <- round(S, 1)

  # finds the bracket that matches S
  bracket <- which(S <= df$cut_sum)[1]

  return(bracket)
}

# test function
print(get_bracket(0.5))
```

```
## [1] 2
```

Task 4

(4 points)

Write a function that takes an argument for the fraction of quarterly quota S reached by the salesperson and returns the dollar amount earned C.

This function must use `df` and `get_bracket()` as globals.

Do not use for loops in completing this task or the next. Instead, let your data frame do the work. In your function, match the amount earned to the appropriate row in your first data frame to get the information needed to computer the answer.

Hints:

You can compute the result using only the data for the row associated with the bracket.

Try to emulate the formula in the instructions, where $C = a + b$, and a is the commission earned from the lower brackets and b is the commission earned on the quota Q reached in the current bracket.

The value of a can be computed by subtracting the payout for the bracket from the cumulative sum of the payout.

There are various ways to compute the value of b. One way is to figure out how much of the earnings apply to the current bracket and then multiply that by the payout percentage. Another is to emulate the example — compute the percentage of S that applies to the bracket and multiply this by Q and the payout percentage.

Note, finally, that you may not need all of the columns in `df` to compute C.

Test your function by passing it a value of $.5$ and make sure it returns a value of 8550.

```
 commission <- function(S) {
  # Ensure S is rounded to 1 decimal place
  S <- round(S, 1)

  # Find the bracket number
  bracket <- get_bracket(S)

  # Compute the amount of quota reached
  sales_amt <- S * quota

  # Compute commission
  if (bracket == 1) {
    C <- sales_amt * df$payout_per[bracket]
  } else {
    # Commission from previous brackets
    a <- df$payout_sum[bracket - 1]

    # Remaining commission in the current bracket
    b_amt <- sales_amt - df$amt_sum[bracket - 1]
    b <- b_amt * df$payout_per[bracket]

    C <- a + b
  }

  return(C)

}
# Test function with S = 0.5
print(commission(0.5))
```

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

Task 5

(2 points)

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

Put the results of your function in a second data frame called `df2` with columns for percent of quota earned S and commission C for that amount.

Display `df2` in a cell.

The result should look like this:

```
     S     C
1  0.1  1575
2  0.2  3150
3  0.3  4725
4  0.4  6300
5  0.5  8550
6  0.6 10800
7  0.7 13050
8  0.8 15975
9  0.9 18900
10 1.0 22500
```

```
# Generate sequence from 0.1 to 1.0
S_values <- seq(0.1, 1.0, by = 0.1)

# Compute commission for each S
C_values <- sapply(S_values, commission)

# Create the data frame df2
df2 <- data.frame(S = S_values, C = C_values)

# Display df2
print(df2)
```

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