Conjoint Analysis using customers preferences dataset

2025-01-18

```
getwd()
```

```
## [1] "C:/Users/sjh50/OneDrive/문서/UC Davis/Winter/Advanced stat/Class_1/Class_2"
```

```
setwd("C:/Users/sjh50/OneDrive/문서/UC Davis/Winter/Advanced stat/Class_1/Class_2")
```

```
# Load necessary libraries
library(ggplot2)
library(readxl)
library(dplyr)
```

```
##
## 다음의 패키지를 부착합니다: 'dplyr'
```

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

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

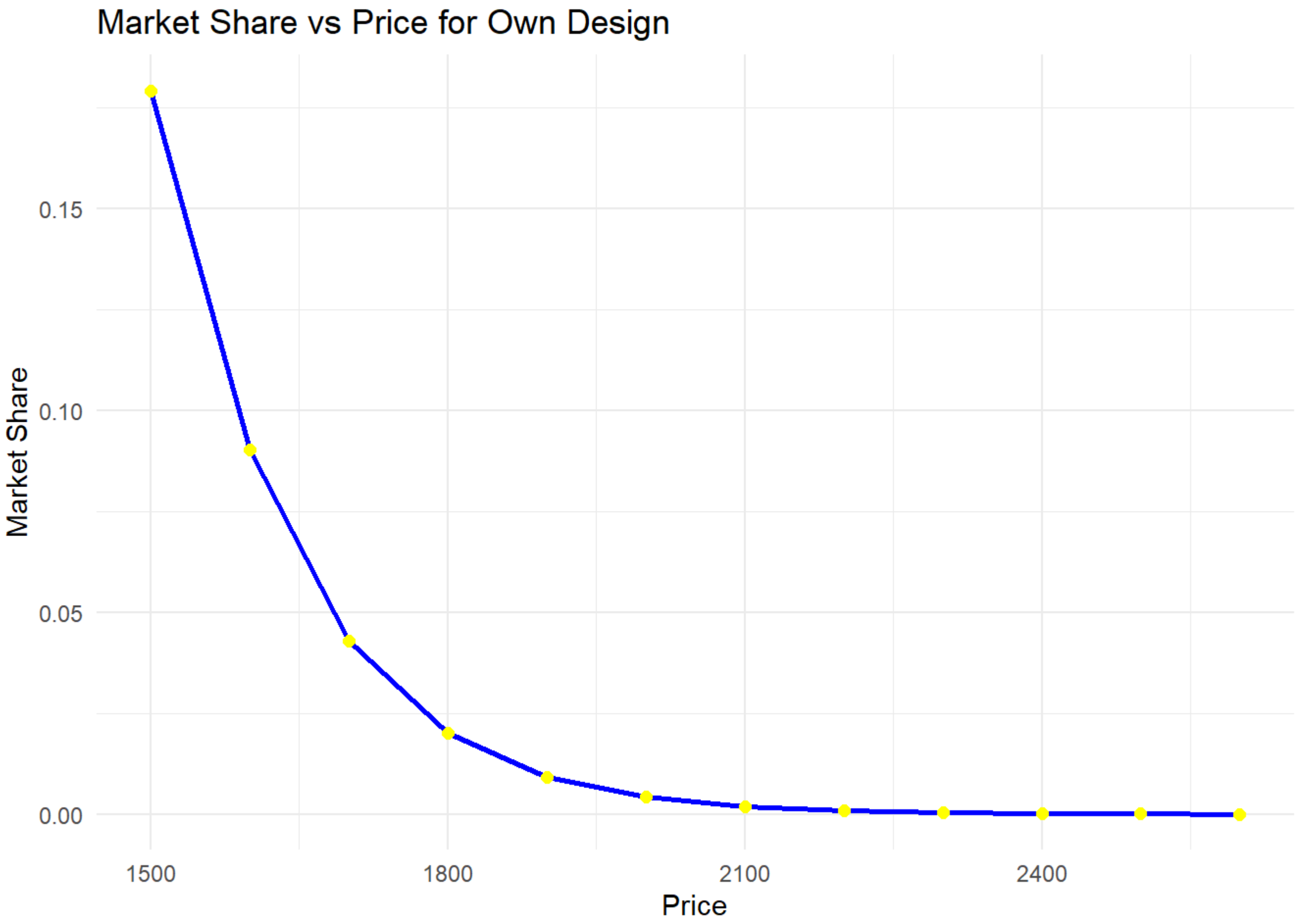
```
# Load the Preferences 2025 dataset
data <- read_excel("C:/Users/sjh50/OneDrive/문서/UC Davis/Winter/Advanced stat/Class_1/Class_2/
Preferences 2025.xlsx")

# Rename columns in Preferences 2025 dataset
data <- data %>%
  rename(
    ProfileNos = `Profile Nos`,
    Profiles = `Profiles`,
    PreferenceRank = `Preference Rank`,
    Screen75Inch = `Screen 75 inch`,
    Screen85Inch = `Screen 85 inch`,
    Resolution4K = `Resolution 4K = 1`,
    Sony = `Sony = 1`,
    PriceLowHigh = `Price ₩r₩n(low = 0; high =1)`
  ) %>%
  mutate(
    PreferenceRank = as.numeric(PreferenceRank),
    Screen75Inch = as.numeric(Screen75Inch),
    Screen85Inch = as.numeric(Screen85Inch),
    Resolution4K = as.numeric(Resolution4K),
    Sony = as.numeric(Sony),
    PriceLowHigh = as.numeric(PriceLowHigh)
  )

# View the imported data
head(data)
```

```
## # A tibble: 6 × 8
##   ProfileNos Profiles      PreferenceRank Screen75Inch Screen85Inch Resolution4K
##        <dbl> <chr>                  <dbl>        <dbl>        <dbl>        <dbl>
## 1         19 75, 1K, Shar…              1            1            0            0
## 2          6 65, 4K, Sony…              2            0            0            1
## 3          9 65, 1K, Shar…              3            0            0            0
## 4         12 65, 4K, Shar…              4            0            0            1
## 5          3 65, 1K, Sony…              5            0            0            0
## 6         13 75, 1K, Sony…              6            1            0            0
## # ℹ 2 more variables: Sony <dbl>, PriceLowHigh <dbl>
```

```
# Define the conjoint analysis function
conjoint_analysis <- function(data, own_design, competitor_A, competitor_B, cost_per_feature, value_per_feature) {
  # Partworth estimation
  lm_fit <- lm(PreferenceRank ~ Screen75Inch + Screen85Inch + Resolution4K + Sony + PriceLowHigh, data = data)
  estimates <- coef(summary(lm_fit))
  # Print estimates
  estimates
  partworths <- coef(lm_fit)
  se <- estimates[, "Std. Error"]
  tval <- estimates[, "t value"]
  # Print partworths for Preference 2025 dataset
  partworths
  # Print standard error
  se
  # Print t-value for each coefficient
  tval

  # Attribute importance
  ranges <- c(
    max(partworths["Screen75Inch"], partworths["Screen85Inch"], 0) - min(partworths["Screen75Inch"], partworths["Screen85Inch"], 0),
    max(partworths["Resolution4K"], 0) - min(partworths["Resolution4K"], 0),
    max(partworths["Sony"], 0) - min(partworths["Sony"], 0),
    max(partworths["PriceLowHigh"], 0) - min(partworths["PriceLowHigh"], 0)
  )
  # Print ranges
  ranges

  importance <- ranges / sum(ranges) * 100
  # Print importance
  importance




  # 여기서부터 다시 체크
  # Willingness to pay
  dollars_per_util <- abs(cost_per_feature[["Price"]] / partworths["PriceLowHigh"])
  WTP <- c(
    Screen85Inch = partworths["Screen85Inch"] * dollars_per_util,
    Resolution4K = partworths["Resolution4K"] * dollars_per_util,
    Sony = partworths["Sony"] * dollars_per_util
  )
  # Print Willingness to Pay
  WTP

  # Market share prediction
  utility <- function(design) {
    sum(c(
      design["Intercept"] * partworths["(Intercept)"],
      design["Screen75Inch"] * partworths["Screen75Inch"],
      design["Screen85Inch"] * partworths["Screen85Inch"],
      design["Resolution4K"] * partworths["Resolution4K"],
      design["Sony"] * partworths["Sony"],
      design["PriceLowHigh"] * partworths["PriceLowHigh"]
    ))
  }

  utilities <- c(
    utility(own_design),
    utility(competitor_A),
    utility(competitor_B)
  )
  # Print utilities for each design
  utilities

  attractiveness <- exp(utilities)
  # Print attractiveness for each design
  attractiveness

  market_shares <- attractiveness / sum(attractiveness)
  # Print market_shares for each design
  market_shares


  # Profit optimization
  prices <- seq(1500, 2600, by = 100)  # Define price range
  market_size <- 100  # Total market size

  # Calculate net cost
  net_cost <- sum(
    own_design["Intercept"] * value_per_feature[["Intercept"]],
    own_design["Screen75Inch"] * value_per_feature[["Screen75Inch"]],
    own_design["Screen85Inch"] * value_per_feature[["Screen85Inch"]],
    own_design["Resolution4K"] * value_per_feature[["Resolution4K"]],
    own_design["Sony"] * value_per_feature[["Sony"]]
  )
  # Print net cost for debugging
  print(paste("Net Cost:", net_cost))

  # Calculate profits dynamically for each price
  profits <- sapply(prices, function(price) {
    # Step 1: Calculate margin
    margin <- price - net_cost

    # Step 2: Calculate updated utilities for each design
    updated_utilities <- c(
      sum(  # Own design with normalized price
        partworths["(Intercept)"] * own_design["Intercept"],
        partworths["Screen75Inch"] * own_design["Screen75Inch"],
        partworths["Screen85Inch"] * own_design["Screen85Inch"],
        partworths["Resolution4K"] * own_design["Resolution4K"],
        partworths["Sony"] * own_design["Sony"],
        partworths["PriceLowHigh"] * normalize_price(price, baseline_cost, price_range)  # Norm
alize price
      ),
      sum(  # Competitor A
        partworths["(Intercept)"] * competitor_A["Intercept"],
        partworths["Screen75Inch"] * competitor_A["Screen75Inch"],
        partworths["Screen85Inch"] * competitor_A["Screen85Inch"],
        partworths["Resolution4K"] * competitor_A["Resolution4K"],
        partworths["Sony"] * competitor_A["Sony"],
        partworths["PriceLowHigh"] * competitor_A["PriceLowHigh"]
      ),
      sum(  # Competitor B
        partworths["(Intercept)"] * competitor_B["Intercept"],
        partworths["Screen75Inch"] * competitor_B["Screen75Inch"],
        partworths["Screen85Inch"] * competitor_B["Screen85Inch"],
        partworths["Resolution4K"] * competitor_B["Resolution4K"],
        partworths["Sony"] * competitor_B["Sony"],
        partworths["PriceLowHigh"] * competitor_B["PriceLowHigh"]
      )
    )


    # Step 3: Calculate attractiveness and market shares
    updated_attractiveness <- exp(updated_utilities)
    updated_market_shares <- updated_attractiveness / sum(updated_attractiveness)

    # Step 4: Calculate sales and profit
    sales <- updated_market_shares[1] * market_size  # Sales for own design
    profit <- sales * margin

    # Debugging: Print intermediate results
    cat("Price:", price, "Sales:", sales, "Margin:", margin, "Profit:", profit, "₩n")

    return(profit)
  })

  # Calculate market shares for own design separately
  own_design_market_shares <- sapply(prices, function(price) {
    # Step 1: Calculate updated utilities for each design
    updated_utilities <- c(
      sum(partworths["(Intercept)"] * own_design["Intercept"],
          partworths["Screen75Inch"] * own_design["Screen75Inch"],
          partworths["Screen85Inch"] * own_design["Screen85Inch"],
          partworths["Resolution4K"] * own_design["Resolution4K"],
          partworths["Sony"] * own_design["Sony"],
          partworths["PriceLowHigh"] * normalize_price(price, baseline_cost, price_range)),
      sum(partworths["(Intercept)"] * competitor_A["Intercept"],
          partworths["Screen75Inch"] * competitor_A["Screen75Inch"],
          partworths["Screen85Inch"] * competitor_A["Screen85Inch"],
          partworths["Resolution4K"] * competitor_A["Resolution4K"],
          partworths["Sony"] * competitor_A["Sony"],
          partworths["PriceLowHigh"] * competitor_A["PriceLowHigh"]),
      sum(partworths["(Intercept)"] * competitor_B["Intercept"],
          partworths["Screen75Inch"] * competitor_B["Screen75Inch"],
          partworths["Screen85Inch"] * competitor_B["Screen85Inch"],
          partworths["Resolution4K"] * competitor_B["Resolution4K"],
          partworths["Sony"] * competitor_B["Sony"],
          partworths["PriceLowHigh"] * competitor_B["PriceLowHigh"])
    )

    # Step 2: Calculate attractiveness and market shares
    updated_attractiveness <- exp(updated_utilities)
```

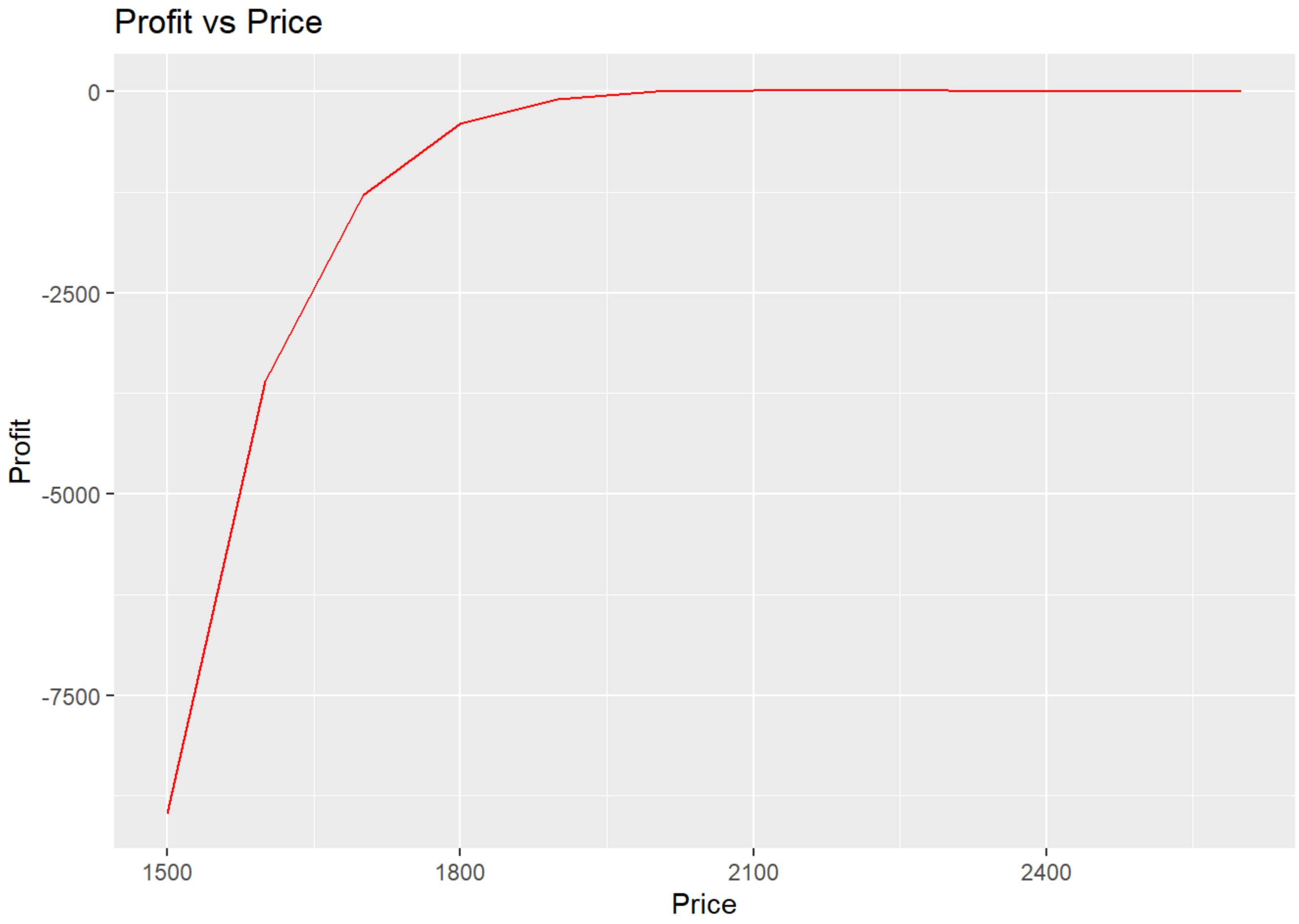
```
    updated_market_shares <- updated_attractiveness / sum(updated_attractiveness)

    # Return market share for own design
    return(updated_market_shares[1])
  })



  # Print the profit results
  print(profits)


  optimal_price <- prices[which.max(profits)]
  # Print optimal_price
  optimal_price

  max_profit <- max(profits)
  # Print max_profit
  max_profit

  # Plots for prices and profits
  profit_plot <- ggplot(data.frame(Price = prices, Profit = profits), aes(x = Price, y = Profi
t)) +
    geom_line(color = "red") +
    labs(title = "Profit vs Price", x = "Price", y = "Profit")


  # Create a data frame for plotting
  plot_data <- data.frame(
    Price = prices,
    OwnDesignMarketShare = own_design_market_shares
  )

  # Plot Market Share vs Price for Own Design
  market_share_plot <- ggplot(plot_data, aes(x = Price, y = OwnDesignMarketShare)) +
    geom_line(color = "blue", size = 1) +
    geom_point(color = "yellow", size = 2) +
    labs(
      title = "Market Share vs Price for Own Design",
      x = "Price",
      y = "Market Share"
    ) +
    theme_minimal()

  # Print the plot
  print(market_share_plot)

  results <- list(
    Partworths = partworths,
    StandardError = se,
    t_values = tval,
    AttributeImportance = importance,
    WillingnessToPay = WTP,
    MarketShares = market_shares,
    OptimalPrice = optimal_price,
    MaxProfit = max_profit,
    ProfitPlot = profit_plot
  )
  return(results)
}

normalize_price <- function(price, baseline_cost, range) {
  (price - baseline_cost) / range
} # normalize price for each function

# Define price range and baseline cost for normalization
price_range <- 2500 - 2000  # Price range
baseline_cost <- 2000       # Baseline cost

# Define design parameters
own_design <- c(Intercept = 1, Screen75Inch = 0, Screen85Inch = 1, Resolution4K = 0, Sony = 0,
PriceLowHigh = normalize_price(2500, baseline_cost, price_range))
competitor_A <- c(Intercept = 1, Screen75Inch = 1, Screen85Inch = 0, Resolution4K = 1, Sony =
1, PriceLowHigh = normalize_price(2500, baseline_cost, price_range))
competitor_B <- c(Intercept = 1, Screen75Inch = 0, Screen85Inch = 1, Resolution4K = 1, Sony =
0, PriceLowHigh = normalize_price(2000, baseline_cost, price_range))

# Define cost and value per feature
cost_per_feature <- list(Price = 500) # set the lowest price as 2000 and the highest price as 2
500
value_per_feature <- list(Intercept = 1000, Screen75Inch = 500, Screen85Inch = 1000, Resolution
4K = 250, Sony = 250)

# Call the function
results <- conjoint_analysis(
  data = data,
  own_design = own_design,
  competitor_A = competitor_A,
  competitor_B = competitor_B,
  cost_per_feature = cost_per_feature,
  value_per_feature = value_per_feature
)
```

```
## [1] "Net Cost: 2000"
## Price: 1500 Sales: 17.92102 Margin: -500 Profit: -8960.511
## Price: 1600 Sales: 9.012935 Margin: -400 Profit: -3605.174
## Price: 1700 Sales: 4.300813 Margin: -300 Profit: -1290.244
## Price: 1800 Sales: 1.998169 Margin: -200 Profit: -399.6338
## Price: 1900 Sales: 0.9165474 Margin: -100 Profit: -91.65474
## Price: 2000 Sales: 0.4179177 Margin: 0 Profit: 0
## Price: 2100 Sales: 0.1900374 Margin: 100 Profit: 19.00374
## Price: 2200 Sales: 0.086307 Margin: 200 Profit: 17.2614
## Price: 2300 Sales: 0.03917478 Margin: 300 Profit: 11.75243
## Price: 2400 Sales: 0.01777687 Margin: 400 Profit: 7.110747
## Price: 2500 Sales: 0.008065906 Margin: 500 Profit: 4.032953
## Price: 2600 Sales: 0.003659553 Margin: 600 Profit: 2.195732
##  [1] -8960.511449 -3605.174075 -1290.243942  -399.633830   -91.654739
##  [6]     0.000000    19.003743    17.261399    11.752434     7.110747
## [11]     4.032953     2.195732
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
# Print results
print(results)
```

```
## $Partworths
##  (Intercept) Screen75Inch Screen85Inch Resolution4K         Sony PriceLowHigh 
##     8.385877     2.600052     4.330155     5.446445     2.083766    -3.951746 
## 
## $StandardError
##  (Intercept) Screen75Inch Screen85Inch Resolution4K         Sony PriceLowHigh 
##    0.2581219    0.2566417    0.2570949    0.2105568    0.2105561    0.2105567 
## 
## $t_values
##  (Intercept) Screen75Inch Screen85Inch Resolution4K         Sony PriceLowHigh 
##    32.488053    10.131059    16.842631    25.866862     9.896492   -18.768086 
## 
## $AttributeImportance
## [1] 27.38505 34.44477 13.17829 24.99189
## 
## $WillingnessToPay
## Screen85Inch.Screen85Inch Resolution4K.Resolution4K                 Sony.Sony 
##                  547.8788                  689.1189                  263.6514 
## 
## $MarketShares
## [1] 8.065906e-05 2.664454e-02 9.732748e-01
## 
## $OptimalPrice
## [1] 2100
## 
## $MaxProfit
## [1] 19.00374
## 
## $ProfitPlot
```


Profit vs Price
0
-2500
-5000
-7500
Profit
1500
1800
2100
2400
Price