450 - Marketing Analytics - Solo 2

R Sangole

Methodology Discussion

Input Data and Data Preparation

The input data for the model was prepared by converting all the attribute variables to dummy variables, which makes 14 columns from the original 5 columns of data. The base levels for each attribute is a vector of -1 instead of the default vector of 0s. Furthermore, 3 interaction variables between price & brand are created. This prepares the X matrix.

The response dataframe of (424 x 36) is prepared into a list (424 long) of reponse vectors (36 x 1). Both these preparations are to allow the data to be used with the `rhierMnlDP` function. No other changes in the data were made.

Table 1: X Dataframe

screen1	screen2	RAM1	RAM2	processor1	processor2	price1	price2	brand1	brand2	brand3
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	1	1	0	-1	-1	0	0	1
0	1	0	1	0	1	0	1	0	1	0

Table 2: Y Vector for 1st respondent, 1st three values

	x
DCM1_1	2
DCM1_2	1
DCM1_3	1

Model Approach

- Hierarchical Bayes Multinomial Logit models are built for two cases - without (Model 1) and with (Model 2) the covariate of prior STC ownership
- Model 1 is used to check price sensitivity to brands using brand:price interaction terms
- Model 2 is used to investigate if STC ownership results in a difference of preferences for any attributes
- Preference shares for two choice scenarios (not part of the original data) are estimated using Model 1

For both models, 100,000 iterations are run, with every 10th iteration saved to the disk. (Iteration runs were tested between 20k, 50k, 100k and 200k. Runs greater than 100k were not seen to add any value to the analysis.)

Modeling Results

Model 1 performances are checked using a few common metrics. The checks for the model are via internal validation, i.e. validation of the predictions against the dataset itself. The accuracy of predictions for the 3

classes is 87%. Compared to a no information rate of 44%, this is a statistically significant result. A Kappa value of 0.81 also indicates that the predictions are significantly different from random guesses.

```
##              choicehat
## choice_actual    1    2    3
##             1 3684  228  292
##             2  418 3751  420
##             3  216  213 6042
```

Prediction across Choice Sets

The figure in the appendix shows the comparison of the Predictions, the Actuals - both in percentages of each of the 36 choice sets. It also shows the differences measured as `Y - Yhat` sorted according to the average error of the choice set predictions. If we consider a 5% cutoff as the acceptable error in the predictions (this needs to be decided by the business), 26 choice sets out of 36 (or 72%) are predicted well. If we observe these carefully, we can see that:

- For the bottom 4 sets, we can see that the respondents have not really selected any particular choice strongly. This is resulted in the highest error rates.
- For the top 20 sets, there is a clear selection of the alternative amongst the 3 choices.
- The highest alternative selections amongst all the 36 choice sets is ~74%

Overall, the model looks to be of high quality, and is suitable for prediction and interpretation.

Deciding the burn-in period

After visually evaluating the runcharts for a few respondents, I have decided to use beta estimates from runs 9000 to 10,000 for all future calculations, keeping 1 to 9000 as the burn-in period for stablization. Refer to burnin figure for a run-chart showing this stablization.

Q3 - Interpret your model results in regard to the attributes' effects on stated preferences.

Raw Choice Counts: Are any alternatives highly preferred by all the respondents?

A quick look into the responses of all 424 respondents pooled together reveals some high level insights. These are simple univariate looks into the data, nevertheless, they offer some insights. Histograms for each of the 5 attributes, where the height of each bar represents the number of respondents who have selected a choice-alternative containing that respective attribute level shows us that:

- Price = 0, i.e. the lowest price of $199 is the most preferred, while Price = 2 ($399) is the least preferred
- Processor = 0, i.e. 1.5Ghz is very strongly the least preferred, with the other two equally preferred
- RAM = 0, i.e. 6GB is less prefered
- For Screen, all three seem to be equally preferred
- For Brand, Brand = 0, i.e. STC seems to have a slight preference with Brand = 2, i.e. Pear following the lead

Interestingly, `choice.ID` which represents the selected alternative amongst each choice is strongly leaning towards the right, with 43% of the respondents selecting choice #3.

For all respondents, do some attributes indicate strong preferences?

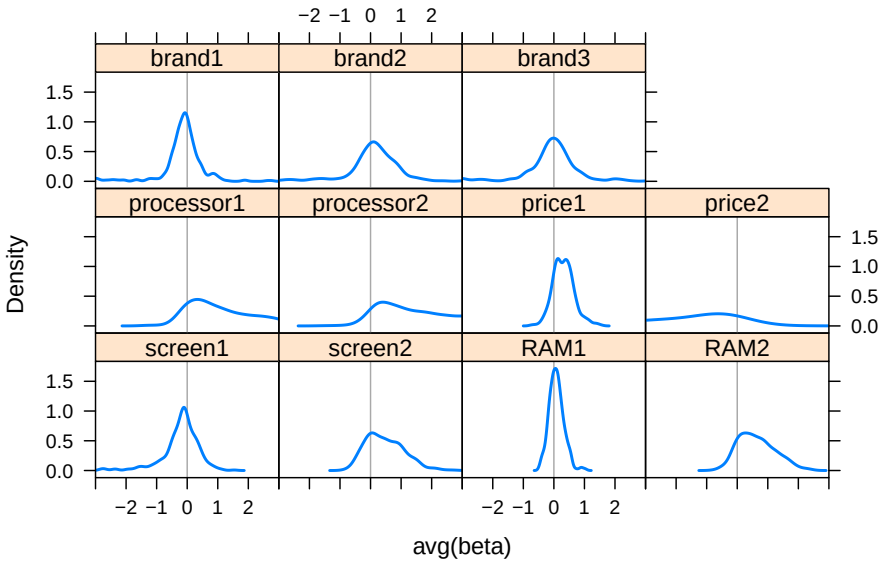
We pool together the average value of the estimated beta coefficients for all the respondents for the simulation runs 9000 to 10000. Thus, we have a matrix of 424 x 14, where each row represents the average beta coefficients for each respondent for the 14 attributes (including 3 interaction terms). The best way to visualize this distribution is using the density plot below, or a box plot as shown in the appendix. Combining these two visuals representations of the data, including the numerical summary of the coefficients in the appendix, we can gain some valuable insights into the minds of the respondents.

Interpreting the average values in table 2, we can see that:

1. Brand: STC is very well positioned in the electronic market. Consumers, on average, perfer Somesong, Pear and Gaggle 48%, 29% and 52% less than they prefer STC, everything else kept constant.
2. Price: Consumers are *extremely* price sensitive. Against a baseline of 199 USD, consumers will prefer a price point of 299 USD only 10% of the times, and virtually never prefer a price of 399 USD.
3. Processor: The conclusions for processor are clear - The higher the processing power, the more the consumers like it. Consumers prefer a 2 GHz processor 30 times more than a 1.5 Ghz processor, while they prefer a 3 GHz processor 40 times more than a 1.5 GHz processor.
4. RAM: 16GB is prefered 2x more than 8GB RAM, while 32GB is prefered 3x more than 8GB RAM, everything else kept constant.
5. Screen: A 7 inch screen is prefered only 7% more than the smaller 5 inch screen. However, a 10 inch screen is prefered 110% over the 5 inch. This is a very strong signal that, on average, consumers prefer much larger screens for their tablets.

However, it's important to remember that this is an interpretation of the average customer sentiment about these attributes, and decisions need to be taken wisely. We can see in the density plot below how much this sentiment varies for some of the attributes. Brand, for example, shows fairly symmetric behavior about zero indicating that respondents do have brand preferences. RAM2 (32GB) and processor2 (2.5 GHz) are very right skewed indicating that overall, the market expects large amounts of both attributes. Similarly for price2 ($399), the sentiment is quite consistently negative. The box plot does show a large number of outliers for all attributes, indicating that such insights are not at all homogeneous.

Avg Beta Values for all respondents for simulations 9000−10,000



Q4 - Study of price sensitivity by brands

By studying the main effects and interaction terms, we can see that STC is very strongly the prefered brand amongst the responders. Interpretation of each value in the table below is: 'By how much do responders prefer STC over the other brand, at the selected price point?' The good news for STC is, irrespective of price, STC is preferred. Thus, there isn't a significant price sensitivity at the macro level. Closely looking at these results graphically as shown in the appendix, no significant interactions can be seen either.

Table 3: Odds Ratios for Brand Price Interaction

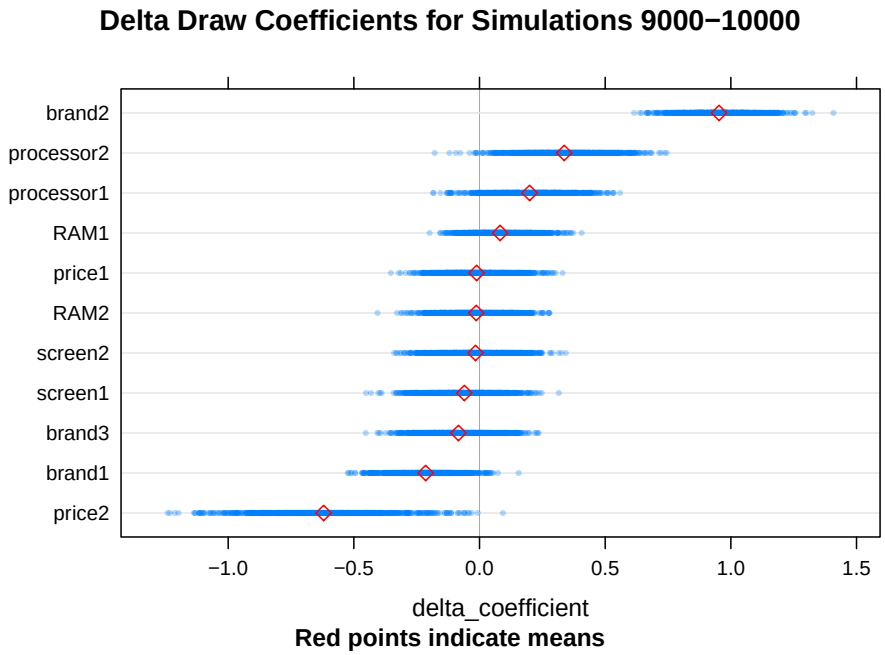
Interaction	USD.199	USD.299	USD.399
STC : Somesong	2.20	1.92	1.67
STC : Pear	1.64	1.41	1.22
STC : Gaggle	2.40	2.08	1.80

Q5 - Impact of ownership of an STC product on the consumer preferences

The impact of previos STC ownership can be measured by introducing the binary variable `vList3` in the input data to the MNL model. The `rhierMnlDP` function outputs a variable called `Deltadraw` which helps us understand which attributes change the most when previous ownership of STC products is true. The graph below shows the values for the final 1000 simulation values (i.e. after removal of burn-in values), with the red points showing the average. What are some key takeaways? Previous ownership of STC makes:

- Respondents highly sensitivity to very high prices ($399)
- Respondents less likely to prefer Somesong (brand1) and Gaggle (brand3), yet, very likely to prefer Pear (brand2). This is a key insight - why would those who own STC want to switch over to Pear?
- Respondents more likely to want 2.5 GHz & 2 GHz processors, and 16 GB RAM
- Respondents less likely to prefer smaller 7 inch screens

Interpretation of the baseline levels for each attribute from this analysis is unknown.



Q6 - Prediction on additional scenarios

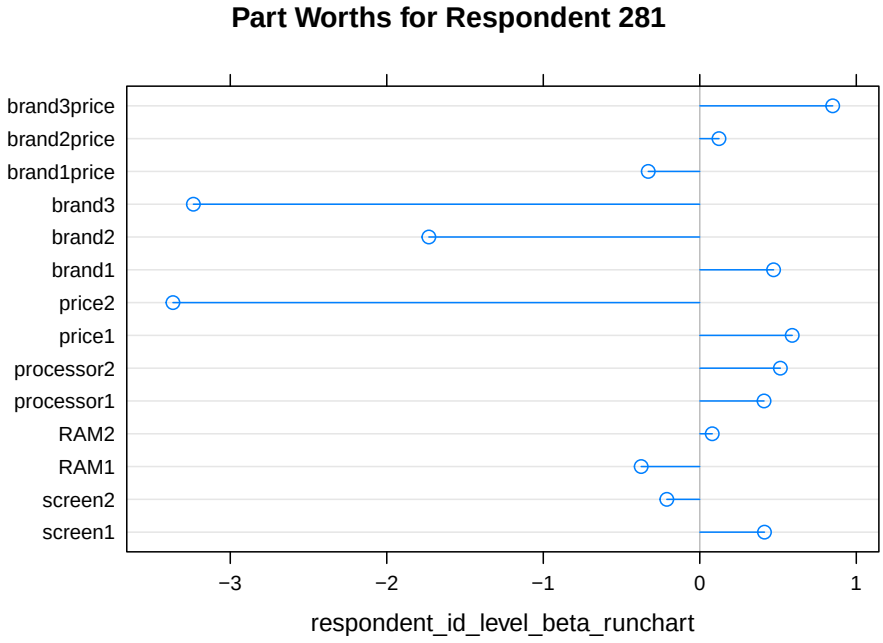
Obee has provided two more choice sets to perform predictions for, as shown in the table below. Applying the 1 predictive model developed using the overall averaged beta coefficients. The predicted probabilities of the choices shown in the table agree with the intuition we've developed in the model above. For choice set 1, a 68% probability is predicted for a largest screen, largest RAM, lowest price and STC brand, just as we expected. However, for choice set 2, the first two alternatives are almost all the lowest configurations for all attributes, and as a result it's roughly a 50-50 chance of selection of either alternative 1 or 2. Not such a strong prediction.

Table 4: Predicted Choices

choice.set	screen	RAM	processor	price	brand	Prob Pred of Choice	Selected Choice
1	2	0	1	0	3	0.07	
1	2	2	1	0	0	0.68	X
1	2	1	1	0	2	0.24	
2	0	0	0	0	0	0.51	X
2	0	1	0	0	3	0.48	
2	1	1	0	2	2	0	

Q7 - Attribute level partworths calculations

Part worths are conceived to be latent values a customer places on levels of an attribute when making choices. Based on the model we've developed, we can calculate the point estimate of average part worth of each attribute, for each respondent. For example, the part worths for respondent 281 is shown in the figure below. This respondent has highest negative sensitivity to high prices, and brands Pear and Gaggle. This type of data can be generated for all respondents.



To calculate this, the following calculation must be run:

1. From the `HBMNL_obj` which is the model object, extract the `betadraw` component first. This will be a 3-D matrix, of dimension (respondent_size x #questions x #simulation_runs_saved). In our example, the size will be (424 x 14 x 10000).

2. Extract the respondent of interest by filtering on the 1st index. For example `HBMNL_obj$betadraw[respondent_id, , ]`. Now, we have 2-D object of dimension (#questions x #simulation_runs_saved).
3. Remove the number of columns based off of the burn off needed. In our example, this reduces (14 x 10000) to (14 x 1000).
4. Calculate the row-means to obtain the part-worth estimates for selected respondent. We will have 14 values corresponding to attributes + interaction terms.
5. Iterate steps 1-4 for desired respondent IDs.

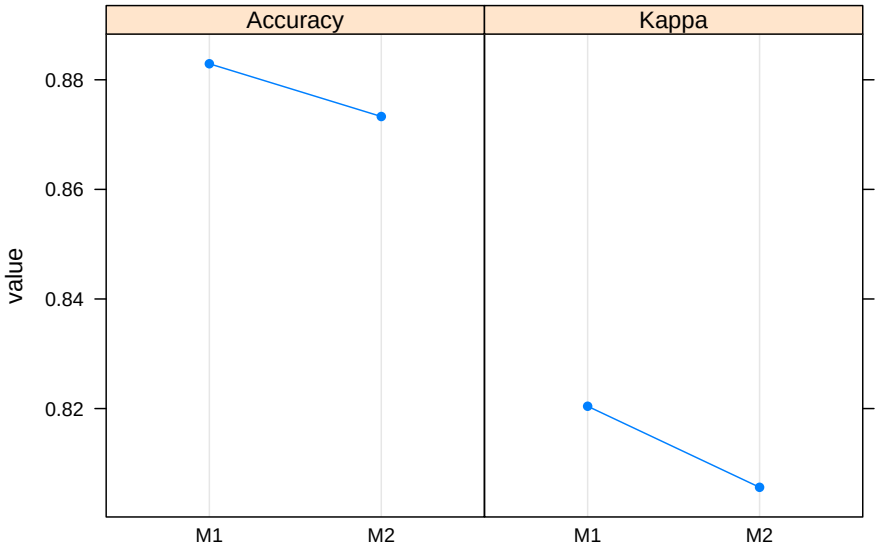
Q8 - Attribute importances for respondents

Since we use dummy coding, one level of each attribute is a base level. The beta coefficients that describe the part worth are with respect to this base level. Attribute importances can be calculated by performing the following algorithm:

1. Calculate each respondents part-worth estimates using the algorithm described in the section above. These will be beta estimates for the dummy variables.
2. For each attribute, add the missing base level. For ex: For {brand1, brand2, brand3}, add brand0 to the dataset.
3. The value for the base variable is equal to `-1* sum(level1, level2,...)`. So now, we have v1 = {brand0, brand1, brand2, brand3}.
4. Calculate odds ratio for each level as follows: v2 = {exp(brand0), exp(brand2), exp(brand3), exp(brand4)}.
5. The odds-ratio indicate how important the levels of the attributes are for that respondent. 1. The effect of moving from the base level to a different level can be computed using the formula exp(level_n - level_base). This can answer the question 'How much more important is level_n compared to level_base?'

Q10 - Comparison of Performance between two models

The covariate indicating prior STC ownership is not significant to the model. A comparison of the performance estimates of both models indicate no difference in the models, when comparing metrics like accuracy or AUC. The AUC for the ROC for both models is 0.863. The accuracy and kappa values are quite similar too, as seen below.



Final Recommendation to STC

Based on all the analysis performed thus far, the recommendations to Obee would to be to develop the following product configuration:

- Price - Lowest (USD 199)
- Brand - STC
- Processor - 2.5 GHz
- RAM - 32 GB
- Screen - 10 inch

The analysis results in Q3 and Q5 support this conclusion. Given the respondents view of STC as the strongest brand, STC is well positioned as a brand to enter the mobile computing market. The price sensitivity was glaringly obvious through all the analyses, and STC should strive to develop at product at the $199 price point only. RAM should be targeted at 32 GB. While existing STC owners are not sensitive between 16 and 32 GB, when all the respondents are seen on aggregate, the response is strongly skewed towards the largest RAM size. The largest screen is also very strongly preferred by a majority of the respondents.

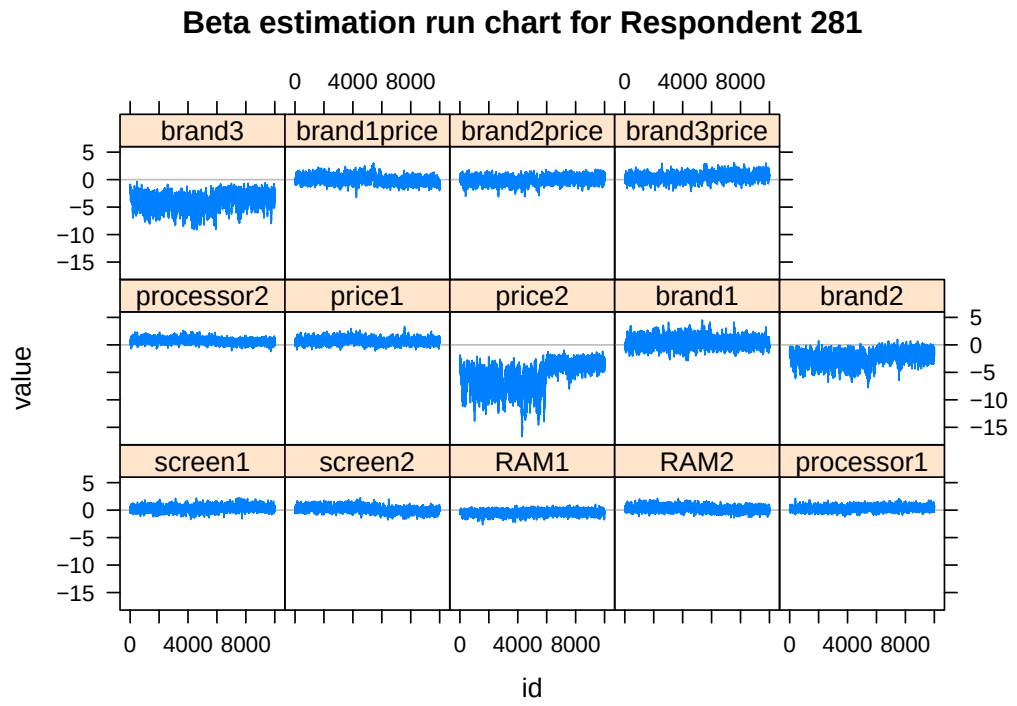
Like any analysis, the outputs are only as good as the input data. A few watchouts for STC to keep in mind are that these analyses are currently only validated on internal data. It would behoove the company to perform some external / hold out style validation as well. Deeper dives into some of the choice sets which predict poorly need to be performed. Perhaps there is some bad data in the overall set. Investigation of other covariates which may add differenciation in product configuration (demographics etc) may add value. There can also some clustering work to be done on the remainder of the dataset not used in this exercise. Simulation 'what if' exercises can answer some questions, and further validation of the selected product can be performed via focus groups or beta-releases into a test market.

Appendix

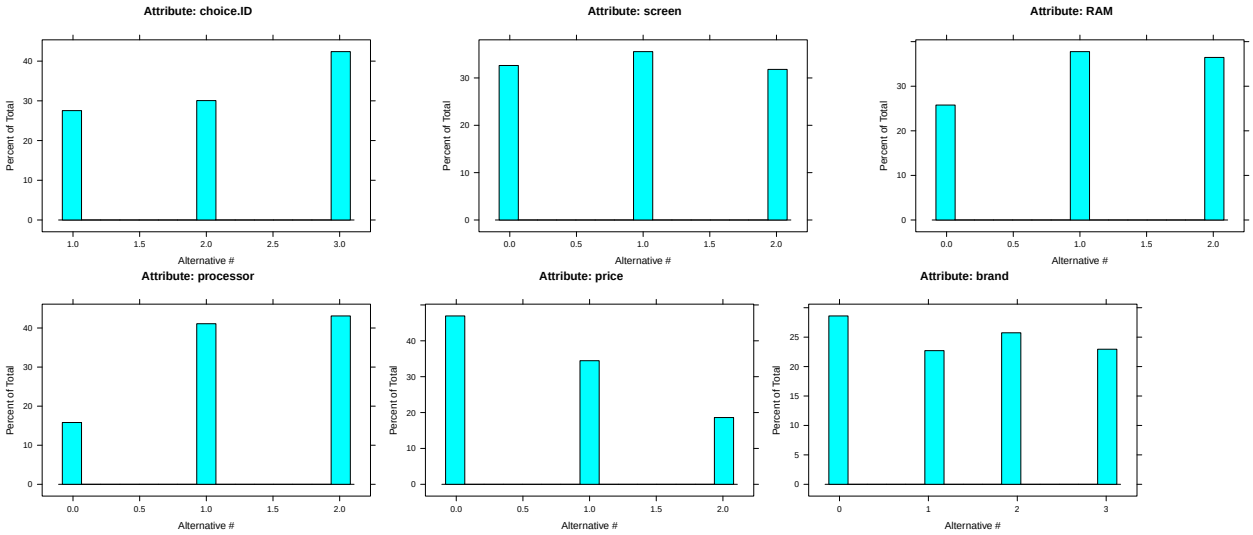
Comparison of the Actual & Predictions from Model 1

	Predictions			Actuals			Difference (Y-Yhat)			Average
	1	2	3	1	2	3	1	2	3	
DCM_28	8%	73%	19%	8%	73%	19%	0%	0%	1%	0%
DCM_10	7%	73%	21%	6%	72%	22%	1%	0%	1%	1%
DCM_7	18%	45%	37%	19%	43%	38%	1%	2%	1%	1%
DCM_19	9%	67%	24%	10%	65%	25%	1%	2%	1%	1%
DCM_1	20%	59%	21%	22%	57%	21%	2%	2%	0%	2%
DCM_12	48%	50%	3%	45%	51%	4%	2%	1%	1%	2%
DCM_9	25%	2%	73%	24%	5%	71%	1%	3%	2%	2%
DCM_18	13%	15%	71%	15%	16%	69%	2%	1%	3%	2%
DCM_22	26%	3%	71%	27%	6%	67%	1%	3%	4%	3%
DCM_17	4%	28%	68%	5%	31%	64%	1%	3%	4%	3%
DCM_24	8%	18%	75%	9%	21%	71%	1%	3%	4%	3%
DCM_16	11%	51%	38%	15%	49%	36%	4%	2%	2%	3%
DCM_25	13%	44%	43%	17%	42%	42%	4%	3%	2%	3%
DCM_29	78%	12%	10%	74%	16%	11%	4%	4%	1%	3%
DCM_26	8%	20%	72%	8%	24%	68%	1%	4%	4%	3%
DCM_34	6%	54%	40%	10%	49%	41%	4%	5%	0%	3%
DCM_3	57%	38%	5%	51%	42%	6%	5%	4%	1%	3%
DCM_27	23%	2%	75%	26%	4%	70%	3%	2%	5%	3%
DCM_14	42%	29%	29%	36%	35%	29%	6%	5%	0%	4%
DCM_15	4%	27%	69%	4%	33%	63%	0%	6%	6%	4%
DCM_35	6%	24%	70%	8%	27%	64%	3%	3%	6%	4%
DCM_8	14%	15%	71%	15%	20%	65%	1%	5%	6%	4%
DCM_20	78%	6%	16%	71%	10%	19%	6%	4%	3%	4%
DCM_31	17%	8%	75%	21%	10%	69%	4%	3%	6%	4%
DCM_13	13%	16%	71%	18%	18%	64%	5%	1%	7%	4%
DCM_2	81%	7%	12%	75%	11%	15%	7%	3%	3%	4%
DCM_6	16%	13%	71%	17%	20%	64%	0%	7%	7%	5%
DCM_33	5%	21%	74%	6%	27%	67%	1%	6%	7%	5%
DCM_4	26%	3%	71%	29%	7%	64%	4%	4%	7%	5%
DCM_32	45%	21%	34%	39%	29%	33%	6%	7%	1%	5%
DCM_5	45%	25%	30%	37%	31%	32%	8%	6%	1%	5%
DCM_36	15%	6%	79%	20%	8%	71%	5%	2%	8%	5%
DCM_23	50%	12%	38%	43%	21%	36%	7%	8%	1%	6%
DCM_30	53%	42%	5%	45%	49%	6%	8%	7%	1%	6%
DCM_11	73%	21%	6%	64%	27%	9%	9%	5%	3%	6%
DCM_21	62%	26%	12%	53%	35%	12%	9%	9%	0%	6%

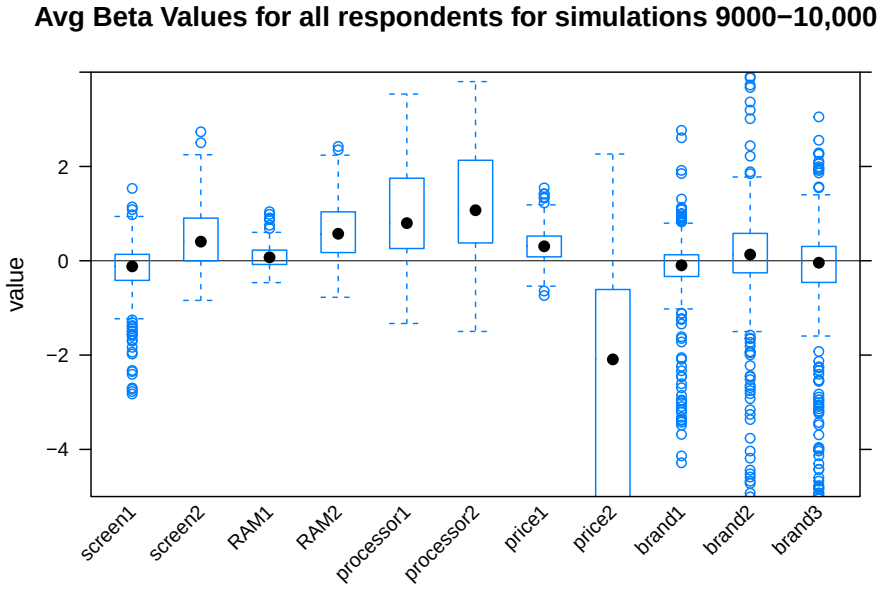
Burn-in Chart for Respondent 281



Histograms of attribute responses for all respondents



Overall beta values for all respondents

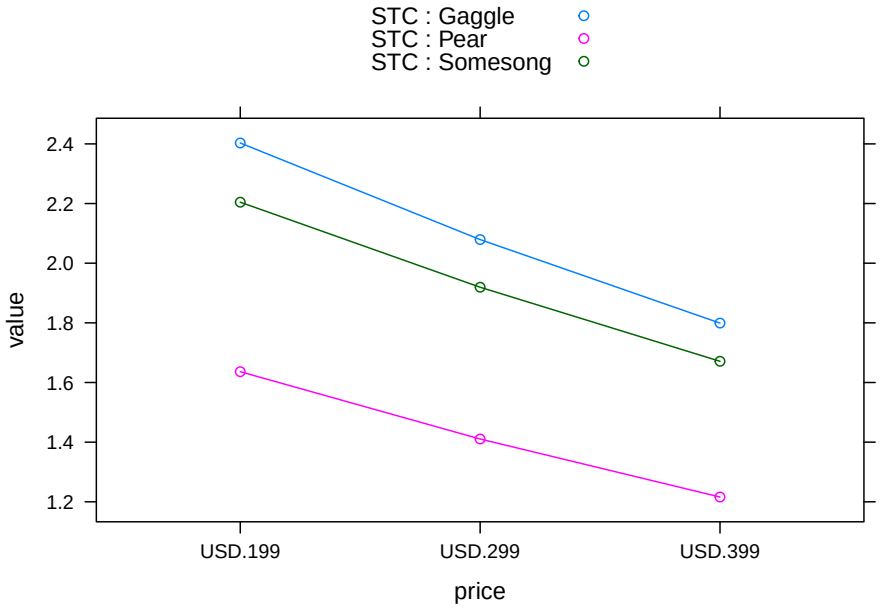


Overall beta values for all respondents

Table 5: Overall beta values for all respondents

variable	attributes	mean	odds.ratio.mean.	preference.over.mean.baseline
brand0	STC	0.43	1.54	NA
brand1	Somesong	-0.22	0.80	0.52
brand2	Pear	0.09	1.09	0.71
brand3	Gaggle	-0.30	0.74	0.48
price0	$199	2.61	13.60	NA
price1	$299	0.31	1.36	0.10
price2	$399	-2.92	0.05	0.00
processor0	1.5Ghz	-2.35	0.10	NA
processor1	2Ghz	1.03	2.80	29.37
processor2	2.5Ghz	1.32	3.74	39.25
RAM0	8gb	-0.72	0.48	NA
RAM1	16gb	0.08	1.09	2.25
RAM2	32gb	0.64	1.90	3.92
screen0	5inch	-0.27	0.76	NA
screen1	7inch	-0.20	0.82	1.07
screen2	10inch	0.47	1.60	2.10

Interaction between price and brand



Code

```
suppressMessages(library('reshape2'))
suppressMessages(library('plyr'))
suppressMessages(library('tidyverse'))
suppressMessages(library('stringr'))
suppressMessages(library('bayesm'))
suppressMessages(library('lattice'))
suppressMessages(library('magrittr'))
suppressMessages(library('bayesm'))
suppressMessages(library('dummies'))

convert_to_effectcodes <- function(df) {
  result_df <- list()
  for (i in 1:ncol(df)) {
    df_i <-
      dummy.data.frame(data = as.data.frame(df[, i]), names = names(df[, i]))
    df_i[df_i[, 1] == 1, ] <- -1
    df_i[, 1] <- NULL
    result_df[[i]] <- df_i
  }
  result_df %>% reduce(bind_cols)
}

plot_allresp_avgbeta <- function(HBMNL_obj, removeinteractions = T,...) {
  betas_from_draws <- apply(HBMNL_obj$betadraw[,,9000:10000],c(1,2),mean)
  betas_from_draws <- as_tibble(betas_from_draws)
  names(betas_from_draws) <- names(xdf)
  if(removeinteractions){
    betas_from_draws %<>% dplyr::select(dplyr::one_of(head(names(xdf),-3)))
  }
  betas_from_draws %>%
    tibble::rownames_to_column(var = 'id') %>%
    reshape2::melt(id = 'id') %>%
    lattice::densityplot(
      ~ value | variable, .,
      panel = function(...) {
        panel.abline(v = 0, col = 'darkgray', lty = 1);
        panel.densityplot(plot.points=F,...)
      },
      main = paste0('Avg Beta Values for all respondents for simulations 9000-10,000'),
      xlab = 'avg(beta)',
      # scales = list(x = list(rot = 45)),
      ...
    )
}

plot_allresp_avgbeta_bwplot <- function(HBMNL_obj, removeinteractions = T,...) {
  betas_from_draws <- apply(HBMNL_obj$betadraw[,,9000:10000],c(1,2),mean)
  betas_from_draws <- as_tibble(betas_from_draws)
  names(betas_from_draws) <- names(xdf)
  if(removeinteractions){
    betas_from_draws %<>% dplyr::select(dplyr::one_of(head(names(xdf),-3)))
```

```
  }
  betas_from_draws %>%
    tibble::rownames_to_column(var = 'id') %>%
    reshape2::melt(id = 'id') %>%
    lattice::bwplot(
      value ~ variable, .,
      panel = function(...) {
        panel.abline(h = 0, col = 'black', lty = 1,lwd=.5);
        panel.bwplot(...)
      },
      main = paste0('Avg Beta Values for all respondents for simulations 9000-10,000'),
      scales = list(x = list(rot = 45)),
      ...
    )
}

plot_runid_bwplots <- function(HBMNL_obj, run_id, h=0, plottype=1, removeinteractions = T,...) {
  betas_from_draws <- as_tibble(HBMNL_obj$betadraw[, , run_id])
  names(betas_from_draws) <- names(xdf)
  if(removeinteractions){
    betas_from_draws %<>% dplyr::select(dplyr::one_of(head(names(xdf),-3)))
  }
  # betas_from_draws <- exp(betas_from_draws)
  # betas_from_draws <- betas_from_draws/(1+exp(betas_from_draws))
  if(plottype==1){
    return(
    betas_from_draws %>%
      tibble::rownames_to_column(var = 'id') %>%
      reshape2::melt(id = 'id') %>%
      lattice::bwplot(
        value ~ variable,
        .,
        panel = function(x, y, ...) {
          panel.abline(h = h, col = 'darkgray', lty = 2);
          panel.bwplot(x, y, ...)
        },
        main = paste0('Beta Values for all respondents for simulation run #', run_id),
        scales = list(x = list(rot = 45)),
        ...
      )
    )
  }
  if(plottype==2){
      return(
        betas_from_draws %>%
        tibble::rownames_to_column(var = 'id') %>%
        reshape2::melt(id = 'id') %>%
        lattice::densityplot(
          ~value| variable,
          .,
          panel = function(...) {
            panel.abline(v = 0, col = 'darkgray', lty = 1);
            panel.densityplot(plot.points=F,...)
```

```
          },
          main = paste0('Beta Values for all respondents for simulation run #', run_id),
          ...
          )
    )
  }
}

plot_respondent_runcharts <-
  function(HBMNL_obj,
           respondent_id,
           plottype = 1:3,
           burnoff = NULL) {
    respondent_id_level_beta_runchart <-
      as_tibble(t(HBMNL_obj$betadraw[respondent_id, , ]))
    names(respondent_id_level_beta_runchart) <- names(xdf)
    respondent_id_level_beta_runchart %<>%
      tibble::rownames_to_column(var = 'id') %>%
      reshape2::melt(id = 'id') %>%
      mutate(id = as.numeric(id))
    if (1 %in% plottype) {
      print(
        xyplot(
          value ~ id | variable,
          respondent_id_level_beta_runchart,
          type = 'l',
          panel = function(...) {
            panel.abline(h = 0, col = 'gray', lty = 1)
            panel.xyplot(...)
          },
          main = paste0('Beta estimation run chart for Respondent ', respondent_id)
        )
      )
    }
    if (2 %in% plottype) {
      if (is.null(burnoff)) {
        nrows <- max(respondent_id_level_beta_runchart$id)
        burnoff <- (0.8 * nrows):nrows
      }
      densityplot(
        ~ value | variable,
        respondent_id_level_beta_runchart[respondent_id_level_beta_runchart$id %in% burnoff, ],
        plot.points = F,
        panel = function(...) {
          panel.abline(v = 0, col = 'gray', lty = 1)

          panel.densityplot(...)
        },
        main = paste0('Density Plot for Respondent ', respondent_id)
      )
    }
    if (3 %in% plottype) {
      respondent_id_level_beta_runchart <- as_tibble(t(HBMNL_obj$betadraw[respondent_id, , ]))
```

```
      names(respondent_id_level_beta_runchart) <- names(xdf)
      if (is.null(burnoff)) {
        nrows <- nrow(respondent_id_level_beta_runchart)
        burnoff <- (0.9 * nrows):nrows
      }
      respondent_id_level_beta_runchart <- respondent_id_level_beta_runchart[burnoff,]
      respondent_id_level_beta_runchart <- colMeans(respondent_id_level_beta_runchart)
       dotplot(respondent_id_level_beta_runchart,
        panel = function(...) {
          panel.abline(v = 0, col = 'gray', lty = 1)
          panel.dotplot(..., type='h')
          panel.dotplot(..., pch=1,cex=1.1,lty=0)
        },
        main = paste0('Part Worths for Respondent ', respondent_id)
      )
    }
  }

main_mat <- read_tsv('../data/stc-dc-task-cbc -v3(1).csv')
head(main_mat)
xdf <- convert_to_effectcodes(main_mat[,c("screen", "RAM", "processor", "price", "brand")])
pricevec = main_mat$price - mean(main_mat$price)
brandsmat <- xdf %>% select(starts_with('brand'))
brands_pricemat <- brandsmat * pricevec
names(brands_pricemat) <- paste0(names(brands_pricemat),'price')
xdf <- as_tibble(cbind(xdf, brands_pricemat))
# xmat has dimensions (108 x 14) == (36 x 3) x 14 == (#questions x #choices/q) x (#dummy variables)
xmat <- as.matrix(xdf)

load('../data/stc-cbc-respondents-v3(1).RData')
ydata <- as_tibble(resp.data.v3) %>%
  select(starts_with('DCM', ignore.case = F))
# ydata is the response from the discrete choice survey
# dimensions are (424 x 36) == (#resp x #ques)

ydata <- na.omit(ydata)

ymat <- as.matrix(ydata)

zowner <- ifelse(is.na(resp.data.v3$vList3),0,1)

lgtdata <- NULL
for (i in 1:424) {
  lgtdata[[i]] <- list(y=ymat[i,],X=xmat)
}

mcmctest <- list(R=100000,keep=10)
Data1 <- list(p=3,lgtdata=lgtdata)
testrun1 <- rhierMnlDP(Data = Data1, Mcmc = mcmctest)
saveRDS(testrun1, file = '../cache/testrun1_250k.Rdata')

zownertest = matrix(scale(zowner,scale = F),ncol = 1)
Data2 <- list(p=3,lgtdata=lgtdata,Z=zownertest)
```

```
testrun2 <- rhierMnlDP(Data = Data2, Mcmc = mcmctest)
saveRDS(testrun2, file = '../cache/testrun2_250k.Rdata')

testrun1 <- read_rds('../cache/testrun1_250k.Rdata')
testrun2 <- read_rds('../cache/testrun2_250k.Rdata')

xdf_long <- main_mat
xdf_long$choice.set <- paste0('DCM1_',xdf_long$choice.set)
ydata_long <- ydata %>% mutate(resp_id = 1:nrow(ydata))
ydata_long <- ydata_long %>% melt(id.vars='resp_id') %>% arrange(resp_id)
ydata_long <- as_tibble(ydata_long)
ydata_long %<>% left_join(xdf_long, by = c('variable'='choice.set'))
ydata_long_filtered <- ydata_long %>%  filter(value==choice.ID) %>% mutate(value=NULL)

plot_allresp_avgbeta(testrun1, xlim=c(-3,3),lwd=2)

read_csv('brand_price_interaction.csv') %>%
  knitr::kable(digits = 2, align = 'c', caption = 'Odds Ratios for Brand Price Interaction')

delta_draws <- testrun2$Deltadraw[9000:10000,1:11]
delta_draws <- as_tibble(delta_draws); names(delta_draws) <- names(xdf)[1:11]
delta_means <- colMeans(delta_draws)
delta_means <- as_tibble(delta_means)
delta_means$attribute <- names(xdf)[1:11]
delta_means$attribute <- factor(x = delta_means$attribute,levels = delta_means %>%
                                  arrange(value) %>% pull(attribute))
delta_means %>% dotplot(attribute~value,.,panel=function(...){panel.abline(v = 0,col='darkgray',lwd=.5)
delta_draws %>% melt(value.name = 'value') %>%
  mutate(variable = factor(variable, levels = levels(delta_means$attribute))) %>%
  dotplot(variable~value,.,xlab='delta_coefficient',panel=function(...){panel.abline(v = 0,col='darkgra
    panel.dotplot(x=delta_means$value,y=delta_means$attribute,col='red',lty = 0,pch=23,cex=1)},
    sub='Red points indicate means',main='Delta Draw Coefficients for Simulations 9000-10000')

read_csv('../reports/new_sc_choices.csv', col_names = T) %>%
  map_df(~str_replace_na(string = .x, replacement = ' ')) %>%
  rename('Prob Pred of Choice'='pred_prob', 'Selected Choice' = 'pred_choice') %>%
  knitr::kable(digits = 0, align = 'c', caption = 'Predicted Choices')

new_scenarious <- read_csv('../data/extra-scenarios(1).csv')
new_scenarious.mat <- as.matrix(new_scenarious)
betameansoverall <- apply(testrun1$betadraw[,,9000:10000],c(2),mean)
betavec <- matrix(betameansoverall,ncol=1,byrow = T)
(new_scenarious.mat %*% betavec) %>%
  matrix(.,ncol=3,byrow=T) %>%
  as_tibble() %>%
  mutate(V1 = exp(V1),
         V2 = exp(V2),
         V3 = exp(V3)) %>%
  rowwise() %>%
  mutate(rsum = V1+V2+V3,
         V1 = V1 / rsum,
         V2 = V2 / rsum,
         V3 = V3 / rsum) -> pchoice_df
```

```r
pchoice_df$choice_hat <- max.col(pchoice_df[,1:3])

plot_respondent_runcharts(testrun1, respondent_id = 281, plottype = 3)

#### Model 1 - No Covariate
beta_means <- apply(testrun1$betadraw[,,9000:10000],c(1,2),mean)
xbeta <- xmat %*% t(beta_means)
xbeta2 <- matrix(xbeta, ncol = 3, byrow = T)
expxbeta2 <- exp(xbeta2)
choice_df1 <- as_tibble(expxbeta2)
choice_df1$rsums <- rowSums(choice_df1)
choice_df1$V1 <- choice_df1$V1/choice_df1$rsums
choice_df1$V2 <- choice_df1$V2/choice_df1$rsums
choice_df1$V3 <- choice_df1$V3/choice_df1$rsums
choice_df1$rsums <- NULL
choice_df1$choicehat <- max.col(choice_df1[,1:3])
choice_df1$choice_actual <- as.vector(t(ydata))
cm1 <- caret::confusionMatrix(xtabs(~choicehat+choice_actual, choice_df1))
#rocTest <- pROC::roc(choice_df1$choice_actual, choice_df2$choicehat, plot=F)
#pROC::auc(rocTest)
#### Model 2 - Covariate Present
beta_means <- apply(testrun2$betadraw[,,9000:10000],c(1,2),mean)
xbeta <- xmat %*% t(beta_means)
xbeta2 <- matrix(xbeta, ncol = 3, byrow = T)
expxbeta2 <- exp(xbeta2)
choice_df1 <- as_tibble(expxbeta2)
choice_df1$rsums <- rowSums(choice_df1)
choice_df1$V1 <- choice_df1$V1/choice_df1$rsums
choice_df1$V2 <- choice_df1$V2/choice_df1$rsums
choice_df1$V3 <- choice_df1$V3/choice_df1$rsums
choice_df1$rsums <- NULL
choice_df1$choicehat <- max.col(choice_df1[,1:3])
choice_df1$choice_actual <- as.vector(t(ydata))
cm2 <- caret::confusionMatrix(xtabs(~choicehat+choice_actual, choice_df1))
# rocTest <- pROC::roc(choice_df1$choice_actual, choice_df2$choicehat, plot=F)
# pROC::auc(rocTest)
cm1$overall %>% bind_rows(cm2$overall) %>% dplyr::select(Accuracy, Kappa) %>%
  bind_cols(Model = c('M1','M2')) %>%
  melt(id.vars='Model',var='Metric') %>% dotplot(value~Model|Metric,.,type='b')

# Performance of the 1st model on predictions of the 36 scenarios
(xmat %*% betavec) %>%
  matrix(.,ncol=3,byrow=T) %>%
  as_tibble() %>%
  mutate(V1 = exp(V1),
         V2 = exp(V2),
         V3 = exp(V3)) %>%
  rowwise() %>%
  mutate(rsum = V1+V2+V3,
         V1 = V1 / rsum,
         V2 = V2 / rsum,
         V3 = V3 / rsum) %>%
  ungroup() -> pchoice_df
```

```r
pchoice_df$choice_hat <- max.col(pchoice_df[,1:3])
freq_resp <- round(pchoice_df[,1:3]*424,0)
names(freq_resp) <- c('Choice1Freq','Choice2Freq','Choice3Freq')
pchoice_df %<>% bind_cols(freq_resp)
```