Analytics on safety feature data: R

```
rm(list=ls())
safety <- read.csv("safetydata.csv")
# str(safety)
# View(safety)
```

We have 9500 observations of 112 variables. Each customer performs 19 choice tasks with a total of 500 customers.

Case: Customer number (1 to 500)

No: Observation number (1 to 9500)

Task: Task number for a customer (1 to 19)

Each customer has a choice among four packages in each choice task. (stated preference data sheet). For example customer 1 in the first choice taks chooses Ch2.

The variables are the following:

CC - Cruise control - 3 levels

GN - Go notifier - 2 levels

NS - Navigation system - 5 levels

BU - Backup aids - 6 levels

FA - Front park assist - 2 levels

LD - Lane departure - 3 levels

BZ - Blind zone alert - 3 levels

FC - Front collision warning - 2 levels

FP - Front collision protection - 4 levels

RP - Rear collision protection - 4 levels

PP - Parallel park aids - 3 levels

KA - Knee air bags - 2 levels

SC - Side air bags - 4 levels

TS - Emergency notification - 3 levels

NV - Night vision system - 3 levels

MA - Driver assisted adjustment - 4 levels

LB - Low speed breaking assist - 4 levels

AF - Adaptive Front lighting - 3 levels

HU - Head up display - 2 levels

Price - Price - 11 levels (300, 1000, 1500, 2000, 2500, 3000, 4000, 5000, 7500, 10000, 12000)

The variables from "segment" to "income" give demographic information

Ch1 = 1 if package 1 is chosen, 0 otherwise

Ch2 = 1 if package 2 is chosen, 0 otherwise

Ch3 = 1 if package 3 is chosen, 0 otherwise

Ch4 = 1 if package 4 is chosen, 0 otherwise

Choice = Ch1 or Ch2 or Ch3 or Ch4 (depending on which option is chosen)

```
table(safety$Choice)
```

```
##
##  Ch1  Ch2  Ch3  Ch4
## 2165 2404 2136 2795
```

```
# convert to numeric categorical in order to run mlogit model later

safety$Choice <- as.numeric(sub("Ch","",safety$Choice))
table(safety$Choice)
```

```
##
##    1    2    3    4
## 2165 2404 2136 2795
```

segment = segment of car population that individual has

year = year

miles = mileage

night = % of time customer drives at night

gender = gender(Female or Male)

age = age bracket

educ = Education level (college (4 years), Grade School, High School, Postgrad, college(1-3 yrs), vocational school)

region = resident of region (MW, NE, SE, SW, W)

Urb = resident of Rural, Suburban, Urban

income = income

```
which(colnames(safety) == "CC1")
```

```
## [1] 4
```

```
which(colnames(safety) == "Price4")
```

```
## [1] 83
```

Columns 4 to 83 of the safety data frame contains the attribute levels for each of the 4 alternatives (choices) indexed by 1, 2, 3 and 4 respectively. Note the zero level in all cases indicate the attribute is not available with the fourth choice being the NONE option. In general higher levels for attributes correspond to more technically advanced features. We can capture the variables either directly or by adding dummy variables that take a value of 1 or 0 depending on the level.

Developing a logit model for the data

- `dfidx`: This command takes the dataframe where we use the first 12 choice tasks for each customer in our training set (12 out of 19), indicates that the shape of the data frame is wide (where each row is an observation), indicates that the variable indicating the choice made is defined by 'Choice', the separator of the variable name and the alternative name (this helps to guess the variables and the alternative name), varying =c(4:83) (helps indicate the variable indices that are alternative specific)
- `idx` identifies the indices
- `idnames` specifies the name of the index for alternatives (the second one here - `alt`) along with the indices already indicated (first one - `NA`).

This creates a data frame of the long format to which we can use the mlogit package.

```
library(mlogit)
```

```
## Loading required package: dfidx
```

```
##
## Attaching package: 'dfidx'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(dfidx)

# varying = attributes we use to create models
# separator = have to mention this in a wide shape dataset
# ie. what separates an attribute name with an alternative (1, 2, 3, 4) regarding the column name
S <- dfidx(subset(safety, Task <= 12), shape = "wide", choice = "Choice", varying = c(4:83), sep ="", id
# S
head(S)
# str(S)
```

```
M <- mlogit(Choice ~ CC + GN + NS + BU + FA + LD + BZ + FC + FP + RP + PP + KA + SC + TS + NV + MA + LB
# -1 because there is no alternate specific choice
# ASC: all of the factors that are not included in the model?
# 1 2 3 4 are not constantly having the same characteristics in its own alternative, but every alternat
summary(M)
```

```
##
## Call:
## mlogit(formula = Choice ~ CC + GN + NS + BU + FA + LD + BZ +
##     FC + FP + RP + PP + KA + SC + TS + NV + MA + LB + AF + HU +
```

```
##     Price - 1, data = S, method = "nr")
##
## Frequencies of alternatives:choice
##       1       2       3       4
## 0.22850 0.25500 0.23567 0.28083
##
## nr method
## 4 iterations, 0h:0m:1s
## g'(-H)^-1g = 2.05E-07
## gradient close to zero
##
## Coefficients :
##        Estimate Std. Error  z-value  Pr(>|z|)
## CC     0.108533   0.020754   5.2295 1.699e-07 ***
## GN     0.067709   0.029925   2.2626 0.0236592 *
## NS     0.021263   0.013039   1.6307 0.1029606
## BU     0.026681   0.010931   2.4408 0.0146541 *
## FA     0.052062   0.029611   1.7582 0.0787097 .
## LD     0.057242   0.020777   2.7550 0.0058689 **
## BZ     0.040601   0.020933   1.9395 0.0524345 .
## FC     0.056979   0.029856   1.9084 0.0563338 .
## FP     0.008004   0.015974   0.5011 0.6163261
## RP     0.058425   0.029830   1.9586 0.0501622 .
## PP     0.038776   0.021007   1.8459 0.0649103 .
## KA     0.122004   0.029667   4.1124 3.915e-05 ***
## SC     0.044948   0.015805   2.8439 0.0044564 **
## TS     0.082892   0.020954   3.9560 7.622e-05 ***
## NV     0.036925   0.020756   1.7790 0.0752422 .
## MA     0.058989   0.015952   3.6980 0.0002173 ***
## LB     0.032066   0.016020   2.0016 0.0453234 *
## AF     0.020830   0.020980   0.9929 0.3207739
## HU     0.067075   0.029949   2.2396 0.0251138 *
## Price -0.203617   0.005830 -34.9255 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-Likelihood: -7567.9
```

Log-likelihood at optimality $LL(\hat{\beta}) = -7567.9$

p = 20

$AIC = -2LL(\hat{\beta}) + 2(parameters) = 15175.8$

500 customers * 12 tasks = 6000 observations

Likelihood Ratio (McFadden Index) $= 1 - \frac{LL(\hat{\beta})}{LL(0)} = 1 - \frac{-7567.9}{6000 \log 1/4} = 1 - \frac{-7567.9}{-8317.76} = 0.09$

Results indicate that CC (cruise control), KA (knee air bags), TS (emerging notification), MA (driver adjusted assessment), Price are very significant at 0.001 level. As should be expected, Price has a negative co-efficient.

LD (lane departure) and SC (side air bags) are significant at 0.01 level.

The non price coefficients have positive signs indicating that these are valued (higher levels of safety features).

Willingness to Pay

Ratio of β_j coefficients can be used to estimate the willingness to pay for a particular attribute.

Suppose β_1 is the coefficient for attribute 1 and β_2 is the coefficient of attribute 2 (Price). Note that β_2 will be typically negative since more the price, lesser the utility. Say β_1 is positive.

$U = \beta_1 x_1 + \beta_2 x_2 + \dots$

Assume unit change (increase) in attribute 1.

$U = \beta_1(x_1 + 1) + \beta_2(x_2 + \Delta) + \dots$

Then $\Delta = \frac{-\beta_1}{\beta_2}$. Δ is the willingess to pay for unit increase in attribute.

```
wtp <- as.numeric(- M$coefficients["CC"] / M$coefficients["Price"])
wtp
```

```
## [1] 0.533024
```

Example, willingness to pay for cruise control

$WTP = \frac{0.1085}{0.2036} = 0.5330.$

To check the correct predictions in a sample is simply based on the maximum predicted choice probability:

```
# actual choices made (ie. true labels)
ActualChoice <- subset(safety, Task <= 12)[,"Choice"]

# predict on the training data to see how well our model choose
# prediction based on the model for each observation
P <- predict(M, newdata = S)
# returns probabilities of choosing each alternative (ie. probabilities of all alternatives are calcula

# returns the alternative with maximum probability
PredictedChoice <- apply(P, 1, which.max)
```

ActualChoice keeps track of actual observed choice. P predicts the choice probability for the given dataset. The apply function applies to the matrix P, across rows (indexed by 1), the function of which.max (identifies index that is maximum).

```
Tabtrain <- table(PredictedChoice, ActualChoice)
Tabtrain
```

```
##                ActualChoice
## PredictedChoice   1   2   3   4
##               1 616 273 226 356
##               2 222 647 235 320
##               3 194 258 629 309
##               4 339 352 324 700
```

```
# calculate the number of correct predictions
# along the diagonal is the correct prediction
CorPredTrain <- sum(diag(Tabtrain)) / sum(Tabtrain)
CorPredTrain
```

```
## [1] 0.432
```

Correct predictions = $\frac{616+647+629+700}{6000} = 0.432$

Assuming a complete random choice for predicting the choice would result in 0.25.

```
Test <- dfidx(subset(safety, Task > 12), shape = "wide", choice = "Choice", varying = c(4:83), sep = ""

TestPredict <- predict(M, newdata = Test)

ActualChoice <- subset(safety, Task > 12)[,"Choice"]

PredictedChoice <- apply(TestPredict, 1, which.max)

Tabtest<- table(PredictedChoice, ActualChoice)
Tabtrain
```

```
##                ActualChoice
## PredictedChoice   1   2   3   4
##               1 616 273 226 356
##               2 222 647 235 320
##               3 194 258 629 309
##               4 339 352 324 700
```

```
Tabtest
```

```
##                ActualChoice
## PredictedChoice   1   2   3   4
##               1 376 120  93 176
##               2 119 435  97 227
##               3 118 124 356 190
##               4 181 195 176 517
```

```
CorPredTest <- sum(diag(Tabtest)) / sum(Tabtest)
CorPredTest
```

```
## [1] 0.4811429
```

Correct predictions = $\frac{376+435+356+517}{3500} = 0.481$

Building even more sophisticated models

mixed logit model

- `rpar()` to indicate that there exists random parameters instead of fixed ones
- `"n"` signifies normal distribution
- `print.level` shows the number of simulation steps taken for value to converge

```
M1 <- mlogit(Choice ~ CC + GN + NS + BU + FA + LD + BZ + FC + FP + RP + PP + KA + SC + TS + NV + MA + LI
```

This fits a mixed logit model where the coefficients are treated as random variables. This is captured with rpar(random parameters) argument where 'n' indicates it is modeled as a normal random variable, panel data captures the fact that we have multiple observations per individual.

```r
summary(M1)
```

```
##
## Call:
## mlogit(formula = Choice ~ CC + GN + NS + BU + FA + LD + BZ +
##     FC + FP + RP + PP + KA + SC + TS + NV + MA + LB + AF + HU +
##     Price - 1, data = S, rpar = c(CC = "n", GN = "n", NS = "n",
##     BU = "n", FA = "n", LD = "n", BZ = "n", FC = "n", FP = "n",
##     RP = "n", PP = "n", KA = "n", SC = "n", TS = "n", NV = "n",
##     MA = "n", LB = "n", AF = "n", HU = "n", Price = "n"), panel = TRUE,
##     print.level = TRUE)
##
## Frequencies of alternatives:choice
##       1       2       3       4
## 0.22850 0.25500 0.23567 0.28083
##
## bfgs method
## 19 iterations, 0h:0m:27s
## g'(-H)^-1g = 1.49E-07
## gradient close to zero
##
## Coefficients :
##           Estimate Std. Error  z-value  Pr(>|z|)
## CC        0.188040   0.027937   6.7308 1.687e-11 ***
## GN        0.174136   0.040826   4.2653 1.996e-05 ***
## NS        0.064735   0.016578   3.9049 9.425e-05 ***
## BU        0.052215   0.014015   3.7257 0.0001948 ***
## FA        0.142118   0.040463   3.5123 0.0004443 ***
## LD        0.093061   0.027259   3.4140 0.0006402 ***
## BZ        0.083822   0.027805   3.0146 0.0025729 **
## FC        0.129688   0.040273   3.2203 0.0012808 **
## FP        0.039331   0.020715   1.8986 0.0576108 .
## RP        0.135745   0.039820   3.4089 0.0006522 ***
## PP        0.091961   0.027309   3.3674 0.0007587 ***
## KA        0.218272   0.039789   5.4858 4.117e-08 ***
## SC        0.085990   0.020441   4.2067 2.591e-05 ***
## TS        0.144697   0.027845   5.1964 2.032e-07 ***
## NV        0.075631   0.027172   2.7834 0.0053785 **
## MA        0.099164   0.020329   4.8778 1.072e-06 ***
## LB        0.074986   0.020520   3.6543 0.0002578 ***
## AF        0.055793   0.027647   2.0180 0.0435884 *
## HU        0.116832   0.040147   2.9101 0.0036134 **
## Price    -0.389929   0.010449 -37.3164 < 2.2e-16 ***
## sd.CC     0.250490   0.044076   5.6831 1.323e-08 ***
## sd.GN     0.319575   0.063962   4.9963 5.845e-07 ***
## sd.NS     0.032854   0.032182   1.0209 0.3073198
## sd.BU    -0.043781   0.025051  -1.7477 0.0805201 .
## sd.FA     0.342497   0.064168   5.3375 9.425e-08 ***
## sd.LD    -0.014767   0.049995  -0.2954 0.7677170
## sd.BZ    -0.064215   0.049945  -1.2857 0.1985489
## sd.FC     0.254667   0.066462   3.8318 0.0001272 ***
## sd.FP     0.022831   0.039211   0.5823 0.5603949
## sd.RP     0.259018   0.068476   3.7826 0.0001552 ***
```

```
## sd.PP     0.013939   0.047261   0.2949 0.7680395
## sd.KA     0.086208   0.072018   1.1970 0.2312944
## sd.SC     0.109158   0.036977   2.9521 0.0031562 **
## sd.TS     0.105628   0.048570   2.1748 0.0296485 *
## sd.NV    -0.075541   0.048323  -1.5633 0.1179893
## sd.MA     0.098782   0.036139   2.7334 0.0062687 **
## sd.LB     0.117737   0.035090   3.3553 0.0007927 ***
## sd.AF     0.201897   0.045751   4.4129 1.020e-05 ***
## sd.HU    -0.136795   0.074972  -1.8246 0.0680586 .
## sd.Price  0.365227   0.011497  31.7660 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-Likelihood: -6531.3
##
## random coefficients
##       Min.      1st Qu.      Median        Mean     3rd Qu. Max.
## CC    -Inf  0.019087245  0.18804013  0.18804013  0.35699302  Inf
## GN    -Inf -0.041413858  0.17413594  0.17413594  0.38968574  Inf
## NS    -Inf  0.042575173  0.06473466  0.06473466  0.08689414  Inf
## BU    -Inf  0.022684944  0.05221457  0.05221457  0.08174420  Inf
## FA    -Inf -0.088892853  0.14211754  0.14211754  0.37312794  Inf
## LD    -Inf  0.083101489  0.09306138  0.09306138  0.10302128  Inf
## BZ    -Inf  0.040510212  0.08382225  0.08382225  0.12713428  Inf
## FC    -Inf -0.042082150  0.12968795  0.12968795  0.30145805  Inf
## FP    -Inf  0.023931425  0.03933068  0.03933068  0.05472994  Inf
## RP    -Inf -0.038960253  0.13574479  0.13574479  0.31044983  Inf
## PP    -Inf  0.082559426  0.09196116  0.09196116  0.10136289  Inf
## KA    -Inf  0.160125153  0.21827159  0.21827159  0.27641803  Inf
## SC    -Inf  0.012363994  0.08599023  0.08599023  0.15961646  Inf
## TS    -Inf  0.073452186  0.14469689  0.14469689  0.21594159  Inf
## NV    -Inf  0.024678814  0.07563071  0.07563071  0.12658261  Inf
## MA    -Inf  0.032536295  0.09916353  0.09916353  0.16579076  Inf
## LB    -Inf -0.004426224  0.07498631  0.07498631  0.15439884  Inf
## AF    -Inf -0.080384294  0.05579318  0.05579318  0.19197066  Inf
## HU    -Inf  0.024564951  0.11683168  0.11683168  0.20909840  Inf
## Price -Inf -0.636270704 -0.38992917 -0.38992917 -0.14358764  Inf
```

This estimates for each random coefficient a mean value and a standard deviation.

```
#M$logLik

AICM <- -2 * M$logLik + 2 * 20

#M1$logLik

AICM1 <- -2 * M1$logLik + 2 * 40

as.numeric(AICM)
```

```
## [1] 15175.86
```

```r
as.numeric(AICM1)
```

```
## [1] 13142.59
```

```r
# mixed logit model has smaller AIC value hence this model seems to fit better
```

Log-likelihood = -6531.3

```r
P1 <- predict(M1, newdata = S)
PredictedChoice1 <- apply(P1, 1, which.max)
ActualChoice <- subset(safety, Task <= 12)[,"Choice"]
Tabtrainmixed <- table(PredictedChoice1, ActualChoice)


Tabtrain
```

```
##                ActualChoice
## PredictedChoice   1   2   3   4
##               1 616 273 226 356
##               2 222 647 235 320
##               3 194 258 629 309
##               4 339 352 324 700
```

```r
Tabtrainmixed
```

```
##                 ActualChoice
## PredictedChoice1   1   2   3   4
##                1 530 243 179 288
##                2 203 552 207 256
##                3 173 220 537 236
##                4 465 515 491 905
```

```r
CorPredTrmix <- sum(diag(Tabtrainmixed)) / sum(Tabtrainmixed)
CorPredTrmix
```

```
## [1] 0.4206667
```

Correct predictions = $\frac{530+552+537+905}{6000} = 0.420$

Note that while the log-likelihood for the mixed logit model is better, in terms of predicting by simply looking at highest probability it is worse than MNL in this example.

```r
TestPredict1 <- predict(M1, newdata = Test)
PredictedChoice1 <- apply(TestPredict1, 1, which.max)
ActualChoice1 <- subset(safety, Task > 12)[,"Choice"]
Tabtestmixed = table(PredictedChoice1, ActualChoice1)

Tabtest
```

```
##                 ActualChoice
## PredictedChoice   1   2   3   4
##               1 376 120  93 176
##               2 119 435  97 227
##               3 118 124 356 190
##               4 181 195 176 517
```

```
Tabtestmixed
```

```
##                  ActualChoice1
## PredictedChoice1   1   2   3   4
##                1 331 116  77 137
##                2 102 377  83 180
##                3  98 110 316 149
##                4 263 271 246 644
```

```
CorPredTestmixed <- sum(diag(Tabtestmixed)) / sum(Tabtestmixed)
CorPredTestmixed
```

```
## [1] 0.4765714
```

Correct predictions $= \frac{331+377+316+644}{3500} = 0.4765$

The mixed logit model does a better job of predicting customers who are not interested in choosing any of the offered options compared to MNL.

We can also compare out of sample (test) log-likelihood.

MNL and Mixed Test

```
ActualChT <- subset(safety, Task > 12)
ActChT <- cbind(ActualChT$Ch1, ActualChT$Ch2, ActualChT$Ch3, ActualChT$Ch4)
MNLtestLL <- sum(ActChT * log(TestPredict))
MixedtestLL <- sum(ActChT * log(TestPredict1))
MNLtestLL
```

```
## [1] -4246.414
```

```
MixedtestLL
```

```
## [1] -4302.115
```