Assignment 2

Zongqi Wang

11/5/2019

SETUP

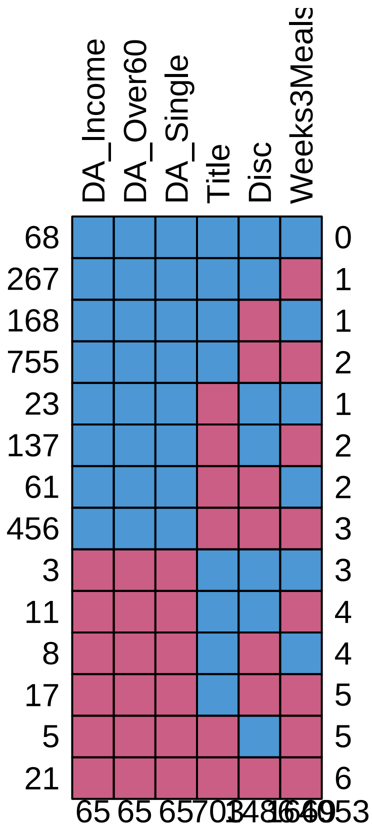
```
library(dplyr)
library(forcats)
library(mice)
library(MASS)
library(corrplot)
library(randomForest)
library(ggplot2)
library(nnet)
library(glmnet)
library(gbm)
library(caret)
library(effects)
source("BCA_functions_source_file.R")
data = read.csv("QK.csv", row.names = 'X')
```

Exploratory analysis

```
#summary(data)
nrow(data)
```

```
## [1] 2000
```

```
#Missing Data
md.pattern( data[,c( "DA_Income", "DA_Over60",
                     "DA_Single", "Title",
                     "Disc", "Weeks3Meals")], rotate.names = TRUE)
```



```
##     DA_Income DA_Over60 DA_Single Title Disc Weeks3Meals
## 68          1         1         1     1    1           1    0
## 267         1         1         1     1    1           0    1
## 168         1         1         1     1    0           1    1
## 755         1         1         1     1    0           0    2
## 23          1         1         1     0    1           1    1
## 137         1         1         1     0    1           0    2
## 61          1         1         1     0    0           1    2
## 456         1         1         1     0    0           0    3
## 3           0         0         0     1    1           1    3
## 11          0         0         0     1    1           0    4
## 8           0         0         0     1    0           1    4
## 17          0         0         0     1    0           0    5
## 5           0         0         0     0    1           0    5
## 21          0         0         0     0    0           0    6
##             65        65        65   703 1486        1669 4053
```

Removing useless data

```
colnames(data)
```

```
##  [1] "custid"        "SUBSCRIBE"     "Disc"          "Title"
##  [5] "LastOrder"     "Pcode"         "DA_Income"     "DA_Under20"
##  [9] "DA_Over60"     "DA_Single"     "NumDeliv"      "NumMeals"
## [13] "MealsPerDeliv" "Healthy"       "Veggie"        "Meaty"
## [17] "Special"       "TotPurch"      "Weeks3Meals"   "Sample"
```

```
#Removing custom ID
data$custid <- NULL
data$Weeks3Meals <- NULL
data$Title <- NULL
head(data)
```

```
##   SUBSCRIBE   Disc  LastOrder   Pcode DA_Income DA_Under20 DA_Over60 DA_Single
## 1      <NA> Senior 2018-01-26 B0V 2H9      57.5        137       105        27
## 2      <NA>   <NA> 2018-01-27 J6R 3P0      73.7         65       186        17
## 3         N   <NA> 2018-01-15 L9N 0L2      53.3        313       176         3
## 4      <NA> Senior 2018-02-14 B1K 1E1     101.9        236        98        39
## 5         N   <NA> 2017-12-18 L3V 1R5      76.6        196        80        34
## 6         N   <NA> 2018-01-10 G0S 1C4      53.6        248       177        50
##   NumDeliv NumMeals MealsPerDeliv Healthy Veggie Meaty Special TotPurch
## 1       23       46             2       9     26    10       1 481.9132
## 2       14       14             1       2      1     0      11 175.9909
## 3       10       10             1       6      1     0       3 117.9338
## 4       47       47             1       2     10    31       4 599.8948
## 5       10       20             2      12      1     7       0 235.5387
## 6       19       38             2      30      0     5       3 505.5448
##       Sample
## 1    Holdout
## 2    Holdout
## 3 Validation
## 4    Holdout
## 5 Validation
## 6 Estimation
```

```
#Removing columns with missing DA_income
data <- data[!is.na(data$DA_Income),]
```

Engineering features

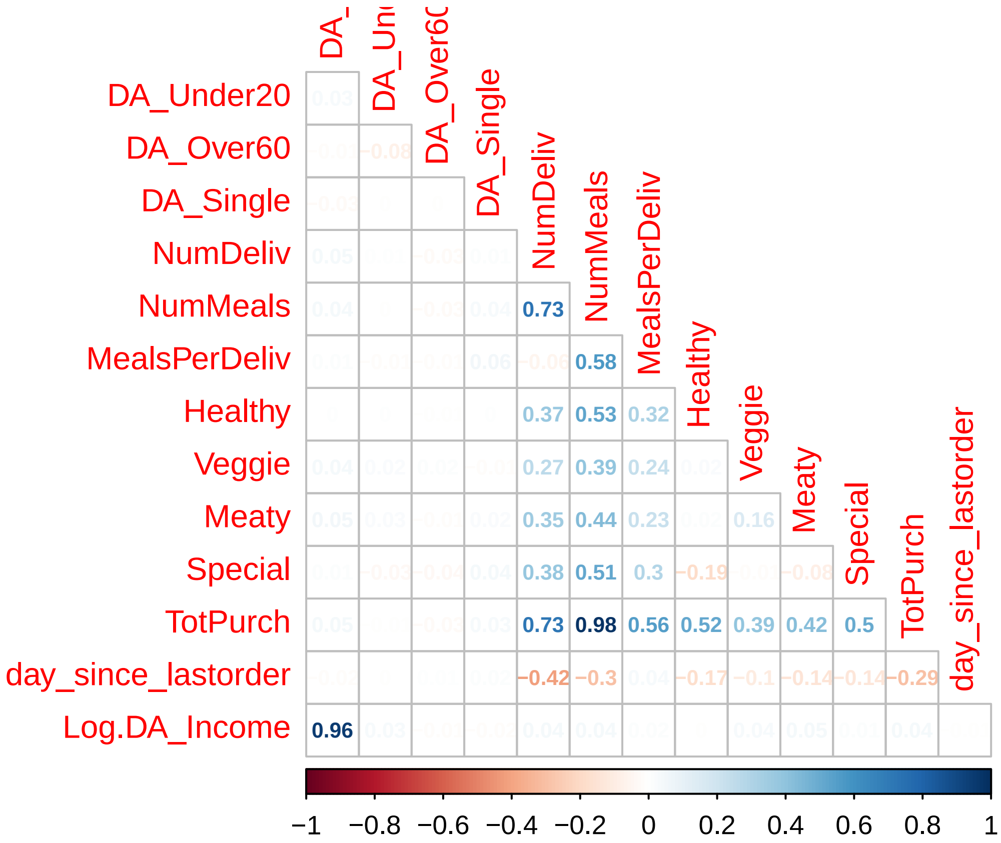
```
#We can't do much with the postal code right now but to convert to provinces
data$Pcode <- NULL
#Number of days passed since last delivery
data$LastOrder = as.Date(data$LastOrder)
data$day_since_lastorder = as.numeric(as.Date("2018-03-05")-data$LastOrder)
data$LastOrder <- NULL

#Log
data$Log.DA_Income <- log(data$DA_Income)

#Changing NA to no discount
data$Disc <- fct_explicit_na(data$Disc, "NoDisc")
```

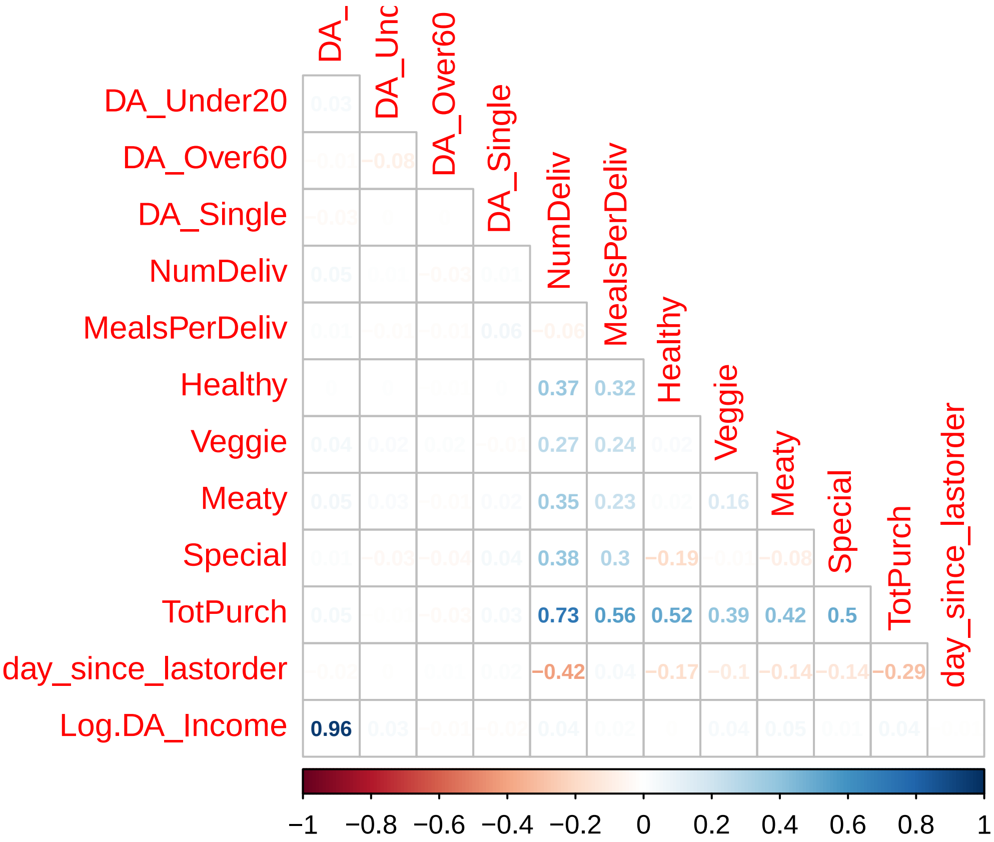
correlation analysis

```r
corrMatrix <- cor(select_if(data, is.numeric))
corrplot(corrMatrix,method="number",type="lower",
diag = FALSE,number.cex = 0.7)
```



NumMeals seems highly correlated with TotPurch

```r
data$NumMeals <- NULL
corrMatrix <- cor(select_if(data, is.numeric))
corrplot(corrMatrix,method="number",type="lower",
diag = FALSE,number.cex = 0.7)
```



Building Models

Seperating data from holdout data

```
holdout = filter(data, data$Sample == "Holdout")

data = filter(data, data$Sample != "Holdout")
#Removing the Sample Column
head(data)
```

```
##   SUBSCRIBE   Disc DA_Income DA_Under20 DA_Over60 DA_Single NumDeliv
## 1         N NoDisc      53.3        313       176         3       10
## 2         N NoDisc      76.6        196        80        34       10
## 3         N NoDisc      53.6        248       177        50       19
## 4         N Senior      79.7        203        97        28       34
## 5         N NoDisc      84.8        108       240        72       27
## 6         Y NoDisc     100.2        178       101        19       50
##   MealsPerDeliv Healthy Veggie Meaty Special TotPurch     Sample
## 1             1       6      1     0       3 117.9338 Validation
## 2             2      12      1     7       0 235.5387 Validation
## 3             2      30      0     5       3 505.5448 Estimation
## 4             2       3      0    64       1 698.1856 Estimation
## 5             2      34      9     1      10 657.4308 Validation
## 6             2       9      0     7      84 1256.2171 Estimation
##   day_since_lastorder Log.DA_Income
## 1                  49      3.975936
## 2                  77      4.338597
## 3                  54      3.981549
## 4                  43      4.378270
## 5                  26      4.440296
## 6                   9      4.607168
```

```
data$Sample <- NULL
```

train test split

```
data.scaled <- as.data.frame(scale(select_if(data, is.numeric)))

train_size = 0.75
smp_size = floor(train_size*nrow(data))

set.seed(123)
train_ind <- sample(seq_len(nrow(data)), size = smp_size)

train <- data[train_ind, ]
test <- data[-train_ind, ]

train.scaled <- data.scaled[train_ind,]
train.scaled$SUBSCRIBE <- data[train_ind, "SUBSCRIBE"]
test.scaled <- data.scaled[-train_ind,]
test.scaled$SUBSCRIBE <- data[-train_ind, "SUBSCRIBE"]
head(train)
```

```
##     SUBSCRIBE   Disc DA_Income DA_Under20 DA_Over60 DA_Single NumDeliv
## 415         N NoDisc      83.8        218       169        35       21
## 463         N NoDisc      54.9        265       203        44       27
## 179         N NoDisc      53.0        121       192        69       12
## 526         N Senior      72.8        249       205        32       14
## 195         N Senior     105.6        182       114        57        8
## 938         N NoDisc      81.5        231       105        31       41
##     MealsPerDeliv Healthy Veggie Meaty Special TotPurch day_since_lastorder
## 415     2.0000000      39      0     0       3 539.8007                  20
## 463     2.0000000      32      3    19       0 639.2157                  46
## 179     3.0000000      10      1     0      25 425.6429                  22
## 526     2.0000000      22      4     2       0 359.7770                 104
## 195     2.0000000       3      0     2      11 177.7431                  68
## 938     0.6097561      12      0     0      13 258.5335                  13
##     Log.DA_Income
## 415      4.428433
## 463      4.005513
## 179      3.970292
## 526      4.287716
## 195      4.659658
## 938      4.400603
```

Training models

```
# Logistic Regression Models
full.mod <- glm(SUBSCRIBE ~ . + NumDeliv:Healthy + NumDeliv:Veggie + NumDeliv:Meaty +
                  NumDeliv:Special, data = train, family = binomial(logit))

step.mod <- stepAIC(full.mod, trace = FALSE)
summary(step.mod)
```

```
##
## Call:
## glm(formula = SUBSCRIBE ~ DA_Under20 + NumDeliv + MealsPerDeliv +
##     Healthy + Veggie + Meaty + Special + day_since_lastorder +
##     Log.DA_Income + NumDeliv:Healthy + NumDeliv:Veggie + NumDeliv:Meaty +
##     NumDeliv:Special, family = binomial(logit), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8563  -0.4660  -0.3222  -0.2068   3.0200
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.248e+01  1.894e+00  -6.586 4.52e-11 ***
## DA_Under20           4.317e-03  1.512e-03   2.855 0.004304 **
## NumDeliv             1.006e-01  2.956e-02   3.402 0.000668 ***
## MealsPerDeliv        3.761e+00  4.195e-01   8.965  < 2e-16 ***
## Healthy             -2.645e-01  3.125e-02  -8.465  < 2e-16 ***
## Veggie              -2.071e-01  5.161e-02  -4.013 5.99e-05 ***
## Meaty               -2.540e-01  3.903e-02  -6.507 7.66e-11 ***
## Special             -2.977e-01  3.271e-02  -9.102  < 2e-16 ***
## day_since_lastorder -1.113e-02  3.904e-03  -2.851 0.004353 **
## Log.DA_Income        1.511e+00  3.716e-01   4.067 4.77e-05 ***
## NumDeliv:Healthy     4.788e-03  7.477e-04   6.403 1.52e-10 ***
## NumDeliv:Veggie      4.926e-03  1.656e-03   2.975 0.002935 **
## NumDeliv:Meaty       3.879e-03  1.030e-03   3.765 0.000166 ***
## NumDeliv:Special     5.886e-03  7.878e-04   7.471 7.97e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 887.53  on 1013  degrees of freedom
## Residual deviance: 611.80  on 1000  degrees of freedom
## AIC: 639.8
##
## Number of Fisher Scoring iterations: 6
```

Lasso

```
myFolds <- createFolds(train$SUBSCRIBE, k = 5)
myControl <- trainControl(
 summaryFunction = twoClassSummary,
 classProbs = TRUE, # IMPORTANT!
 verboseIter = TRUE,
 savePredictions = TRUE,
 index = myFolds
)
model_glmnet <- train(SUBSCRIBE ~ .  + NumDeliv:Healthy + NumDeliv:Veggie + NumDeliv:Meaty + NumDeliv:S
  metric = "ROC",
  method = "glmnet",
  trControl = myControl
)
```

```
## + Fold1: alpha=0.10, lambda=0.02064
## - Fold1: alpha=0.10, lambda=0.02064
## + Fold1: alpha=0.55, lambda=0.02064
## - Fold1: alpha=0.55, lambda=0.02064
## + Fold1: alpha=1.00, lambda=0.02064
## - Fold1: alpha=1.00, lambda=0.02064
## + Fold2: alpha=0.10, lambda=0.02064
## - Fold2: alpha=0.10, lambda=0.02064
## + Fold2: alpha=0.55, lambda=0.02064
## - Fold2: alpha=0.55, lambda=0.02064
## + Fold2: alpha=1.00, lambda=0.02064
## - Fold2: alpha=1.00, lambda=0.02064
## + Fold3: alpha=0.10, lambda=0.02064
## - Fold3: alpha=0.10, lambda=0.02064
## + Fold3: alpha=0.55, lambda=0.02064
## - Fold3: alpha=0.55, lambda=0.02064
## + Fold3: alpha=1.00, lambda=0.02064
## - Fold3: alpha=1.00, lambda=0.02064
## + Fold4: alpha=0.10, lambda=0.02064
## - Fold4: alpha=0.10, lambda=0.02064
## + Fold4: alpha=0.55, lambda=0.02064
## - Fold4: alpha=0.55, lambda=0.02064
## + Fold4: alpha=1.00, lambda=0.02064
## - Fold4: alpha=1.00, lambda=0.02064
## + Fold5: alpha=0.10, lambda=0.02064
## - Fold5: alpha=0.10, lambda=0.02064
## + Fold5: alpha=0.55, lambda=0.02064
## - Fold5: alpha=0.55, lambda=0.02064
## + Fold5: alpha=1.00, lambda=0.02064
## - Fold5: alpha=1.00, lambda=0.02064
## Aggregating results
## Selecting tuning parameters
## Fitting alpha = 0.1, lambda = 0.000206 on full training set
```

```
model_glmnet$results
```

```
##   alpha       lambda       ROC      Sens      Spec      ROCSD      SensSD
## 1  0.10 0.0002064338 0.7701707 0.9472510 0.3913396 0.02163043 0.016022991
## 2  0.10 0.0020643383 0.7581936 0.9704031 0.2888324 0.02304749 0.013432257
## 3  0.10 0.0206433833 0.7073187 0.9844712 0.1878876 0.02550733 0.013480757
## 4  0.55 0.0002064338 0.7691286 0.9457852 0.4053052 0.02347778 0.015270062
## 5  0.55 0.0020643383 0.7598374 0.9692297 0.3043241 0.02245252 0.013578722
## 6  0.55 0.0206433833 0.6938676 0.9874012 0.1490916 0.01965058 0.012661870
## 7  1.00 0.0002064338 0.7662992 0.9416834 0.4254603 0.02882012 0.018542128
## 8  1.00 0.0020643383 0.7617730 0.9663001 0.3089874 0.02271380 0.015525202
## 9  1.00 0.0206433833 0.6868895 0.9926755 0.1211483 0.02395878 0.009431158
##       SpecSD
## 1 0.02556092
## 2 0.03273886
## 3 0.04944757
## 4 0.03253532
## 5 0.03829076
## 6 0.05044900
## 7 0.05289915
## 8 0.03505594
## 9 0.04348879
```

```
#Random Forest model

full.rf <- randomForest(formula = SUBSCRIBE ~ Disc + day_since_lastorder +
                          DA_Income + DA_Under20 + DA_Over60 + DA_Single + NumDeliv + TotPurch +
                          MealsPerDeliv + Healthy + Veggie + Meaty ,
                        data = train,
                        importance = TRUE,
                        ntree = 750, mtry = 3)


rf2 <- randomForest(formula = SUBSCRIBE ~ DA_Income + Log.DA_Income+ DA_Under20 + DA_Single + TotPurch +
                    MealsPerDeliv + Healthy + Veggie + Meaty + day_since_lastorder + NumDeliv:Healthy +
                      NumDeliv:Veggie + NumDeliv:Meaty + NumDeliv:Special,
                    data = train, importance = TRUE,tree = 1000, mtry = 5) # default values


rf3 <- randomForest(formula = SUBSCRIBE ~DA_Income + DA_Under20 + DA_Single + TotPurch +
                    MealsPerDeliv + Healthy + Veggie + Meaty + day_since_lastorder + TotPurch:Healthy +
                      TotPurch:Veggie + TotPurch:Meaty,
                    data = train, importance = TRUE,tree = 1000, mtry = 5) # default values
```

Neural Networks

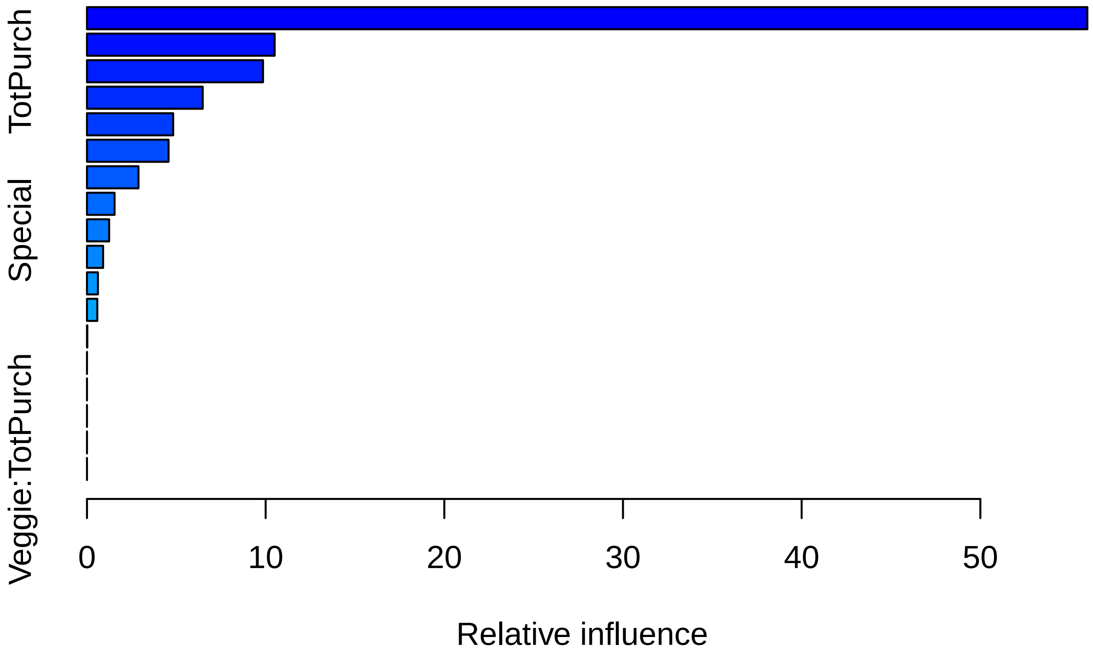
```
nn4 <- Nnet(formula = SUBSCRIBE ~ Disc + day_since_lastorder +
                 DA_Income + DA_Under20 + DA_Over60 + DA_Single + NumDeliv + TotPurch +
                 MealsPerDeliv + Healthy + Veggie + Meaty,
               data = train,
               decay = 0.05,
               size = 4)

nn6 <- Nnet(formula = SUBSCRIBE ~DA_Income + DA_Under20 + DA_Single + TotPurch +
                    MealsPerDeliv + Healthy + Veggie + Meaty + day_since_lastorder + TotPurch:Healthy +
                      TotPurch:Veggie + TotPurch:Meaty,
               data = train,
               decay = 0.05,
               size = 4)
```

Performance Comparison

gradient boosting

```
boost.dat <- train
boost.dat$Binary.Sub <- dplyr::recode(boost.dat$SUBSCRIBE, "Y" = 1, "N" = 0)
boost.dat$SUBSCRIBE <- NULL
gb.mod <- gbm(Binary.Sub ~ .  + NumDeliv:Healthy + NumDeliv:Veggie +
                        TotPurch:Healthy + TotPurch:Veggie,
              data = boost.dat, distribution = "bernoulli", n.trees = 2000, shrinkage = 0.005)
summary(gb.mod)
```



```
##                                     var     rel.inf
## NumDeliv                       NumDeliv 55.98973513
## Veggie                           Veggie 10.50429928
## TotPurch                       TotPurch  9.85415394
## DA_Income                     DA_Income  6.47909777
## day_since_lastorder day_since_lastorder  4.82405090
## DA_Under20                   DA_Under20  4.56565503
## MealsPerDeliv             MealsPerDeliv  2.88359582
## DA_Single                     DA_Single  1.54602309
## Special                         Special  1.24044612
## DA_Over60                     DA_Over60  0.90372377
## Meaty                             Meaty  0.61217135
## Healthy                          Healthy  0.57786372
## Disc                                Disc  0.01918408
## Log.DA_Income              Log.DA_Income  0.00000000
## NumDeliv:Healthy        NumDeliv:Healthy  0.00000000
## NumDeliv:Veggie          NumDeliv:Veggie  0.00000000
## Healthy:TotPurch        Healthy:TotPurch  0.00000000
## Veggie:TotPurch          Veggie:TotPurch  0.00000000
```

```
#Full Model
test$full.mod.pred <- predict(full.mod, test, type = "response")
test$full.mod.pred <- ifelse(percent_rank(test$full.mod.pred)>= 0.6, "yes", "no")
table(test$SUBSCRIBE, test$full.mod.pred)
```

```
##
##      no yes
##   N 190  88
##   Y  13  47
```

```
test.mod <- glm(SUBSCRIBE ~ . + NumDeliv:Healthy + NumDeliv:Veggie + NumDeliv:Meaty + NumDeliv:Special,

test.scaled$full.mod.pred <- predict(test.mod, test.scaled, type = "response")
test.scaled$full.mod.pred <- ifelse(percent_rank(test.scaled$full.mod.pred)>= 0.6, "yes", "no")
table(test.scaled$SUBSCRIBE, test.scaled$full.mod.pred)
```

```
##
##      no yes
##   N 190  88
##   Y  13  47
```

```
# Step Model
test$step.mod.pred <- predict(step.mod, test, type = "response")
test$step.mod.pred <- ifelse(percent_rank(test$step.mod.pred) >= 0.6, "yes", "no")
table(test$SUBSCRIBE, test$step.mod.pred)
```

```
##
##      no yes
##   N 189  89
##   Y  14  46
```

```
# Lasso
test$lasso <- predict(model_glmnet, newdata = test, type = "prob")[,"Y"]
test$lasso <- ifelse(percent_rank(test$lasso)>= 0.6, "yes", "no")
table(test$SUBSCRIBE, test$lasso)
```

```
##
##      no yes
##   N 190  88
##   Y  13  47
```

```r
#Full random forest
test$rf.pred <- predict(full.rf, test, type = "prob")[, 'Y']
test$rf.pred <- ifelse(percent_rank(test$rf.pred) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$rf.pred)
```

```
##
##      no yes
##   N 192  86
##   Y  11  49
```

```r
# baseline random forest
test$rf2.pred <- predict(rf2, test, type = "prob")[, 'Y']
test$rf2.pred <- ifelse(percent_rank(test$rf2.pred) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$rf2.pred)
```

```
##
##      no yes
##   N 192  86
##   Y  14  46
```

```r
# baseline random forest
test$rf3.pred <- predict(rf3, test, type = "prob")[, 'Y']
test$rf3.pred <- ifelse(percent_rank(test$rf3.pred) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$rf.pred)
```

```
##
##      no yes
##   N 192  86
##   Y  11  49
```

```r
# GBM
test$gbm <- predict(gb.mod, test, n.trees = 2000, type = "response")
test$gbm <- ifelse(percent_rank(test$gbm) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$gbm)
```

```
##
##      no yes
##   N 192  86
##   Y  11  49
```

```r
# Nnet 4
test$nnet4.pred <- predict(nn4, test)
test$nnet4.pred <- ifelse(percent_rank(test$nnet4.pred) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$nnet4.pred)
```

```
##
##      no yes
##   N 191  87
##   Y  12  48
```

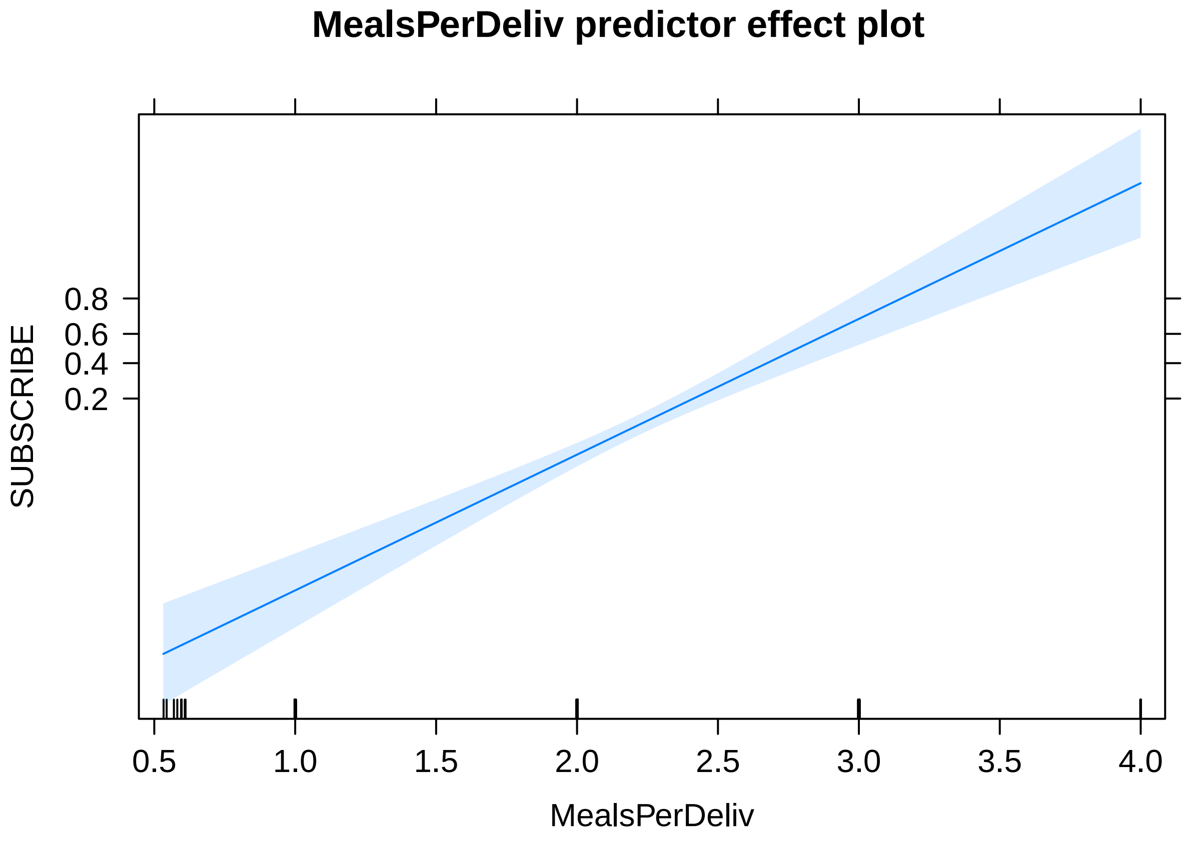
Ensembling

```
test$nnet4.pred <- predict(nn4, test)
test$rf.pred <- predict(full.rf, test, type = "prob")[, 'Y']
test$rf2.pred <- predict(rf2, test, type = "prob")[, 'Y']
test$rf3.pred <- predict(rf3, test, type = "prob")[, 'Y']
test$ensemble <- rowMeans(test %>% dplyr::select(nnet4.pred, rf.pred, rf2.pred, rf3.pred))
test$ensemble <- ifelse(percent_rank(test$ensemble) >=0.6, "yes", "no")
table(test$SUBSCRIBE, test$ensemble)
```

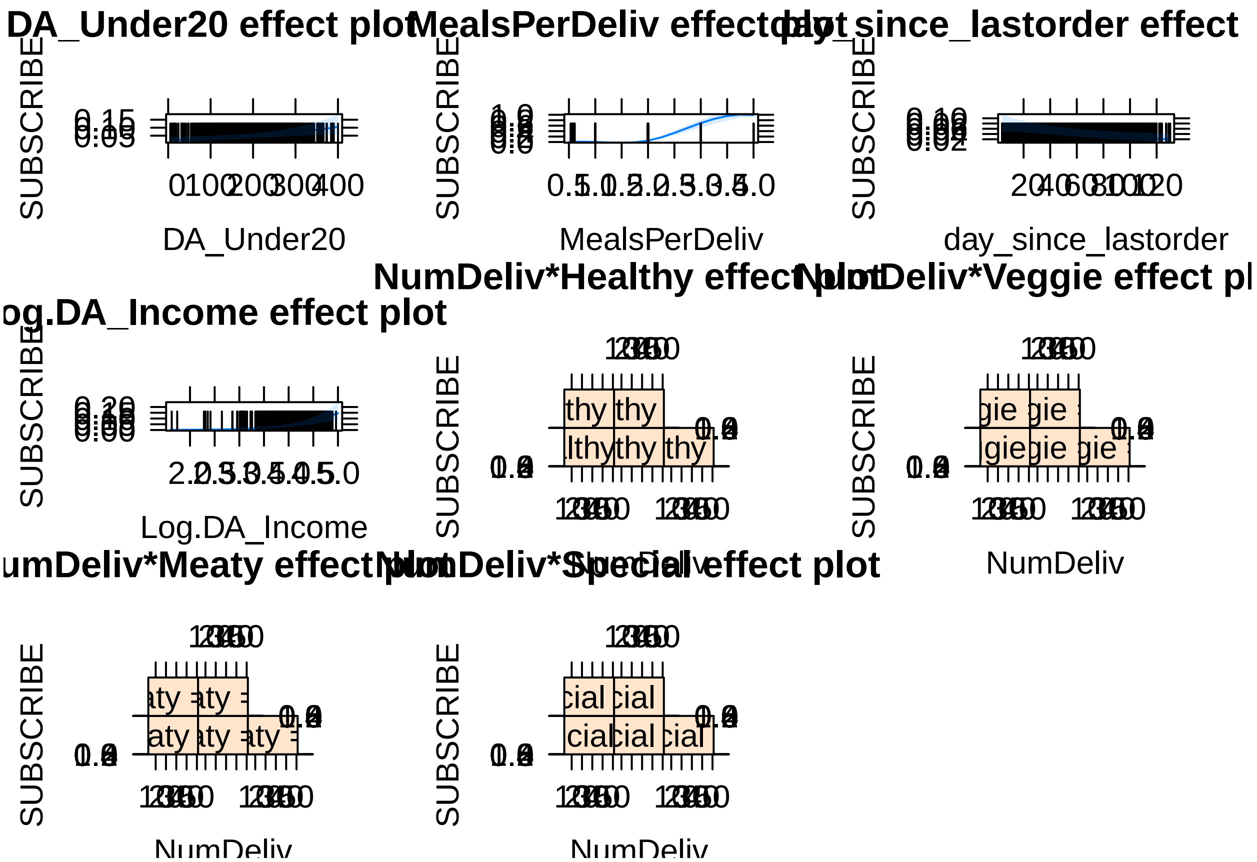
```
##
##      no yes
##   N 191  87
##   Y  12  48
```

Effect Plots

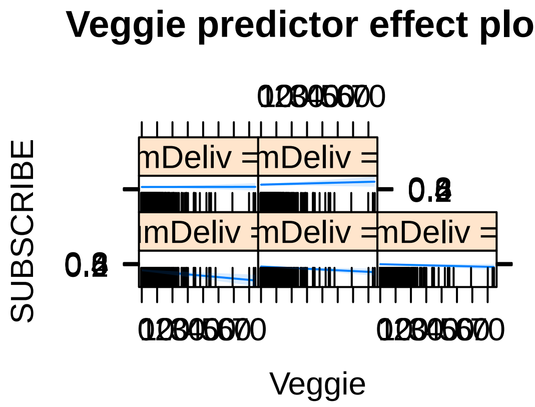
```
plot(predictorEffects(step.mod,"MealsPerDeliv"))
```

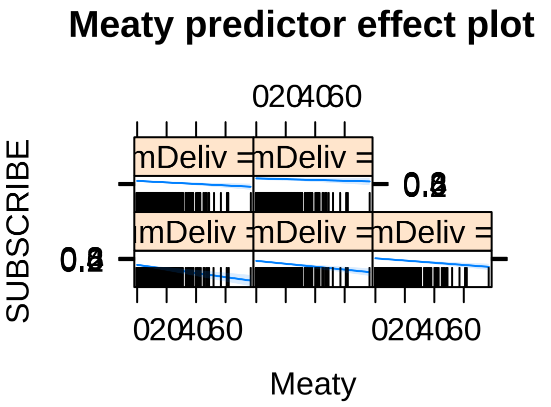


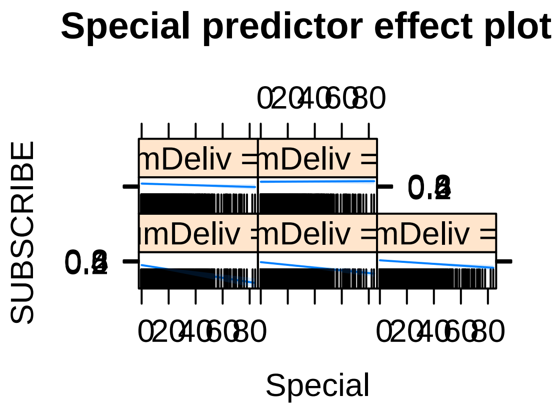
```
plot(allEffects(step.mod), type="response")
```

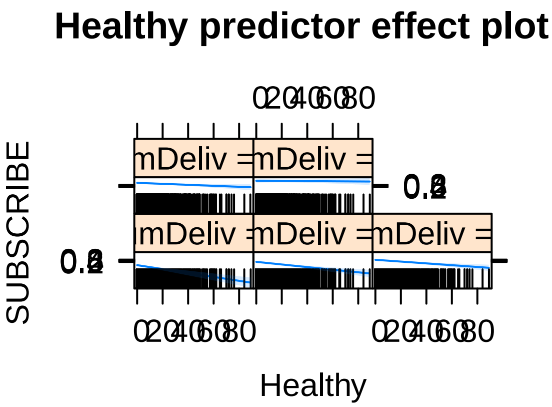


```r
plot(predictorEffects(full.mod,c("Veggie", "Meaty", "Special", "Healthy")))
```



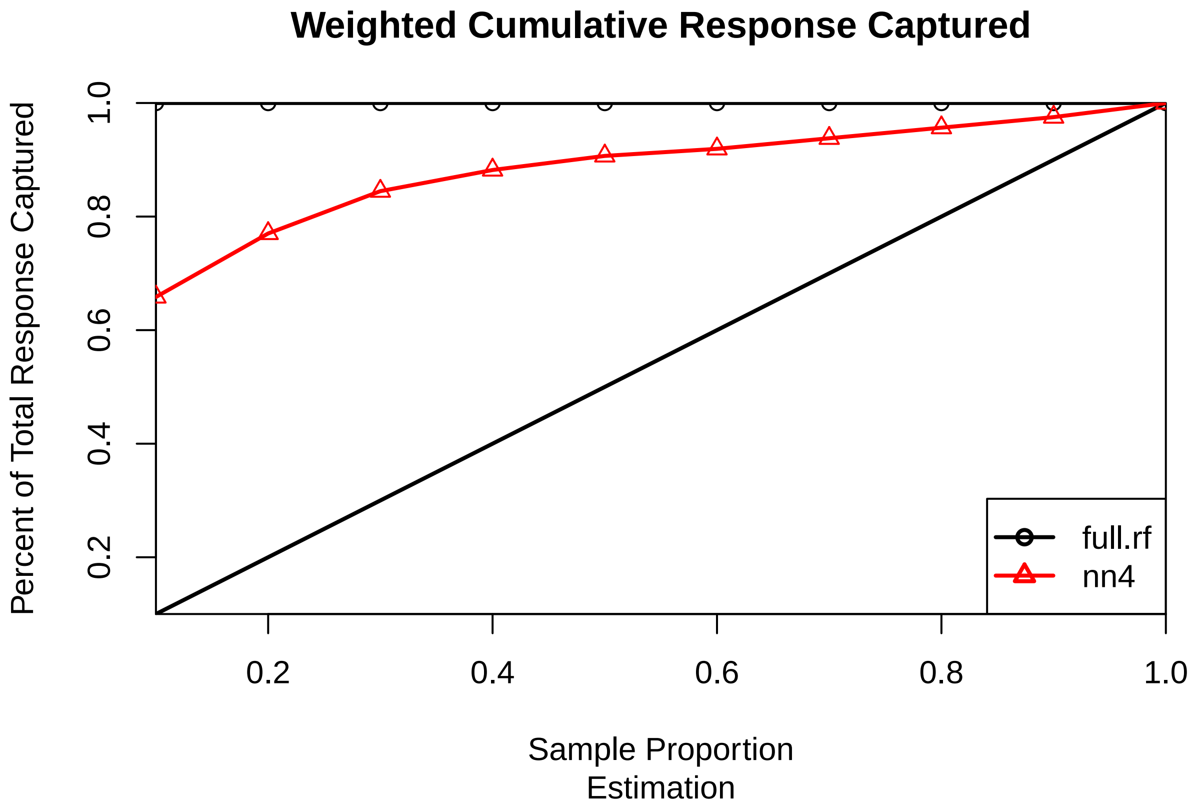




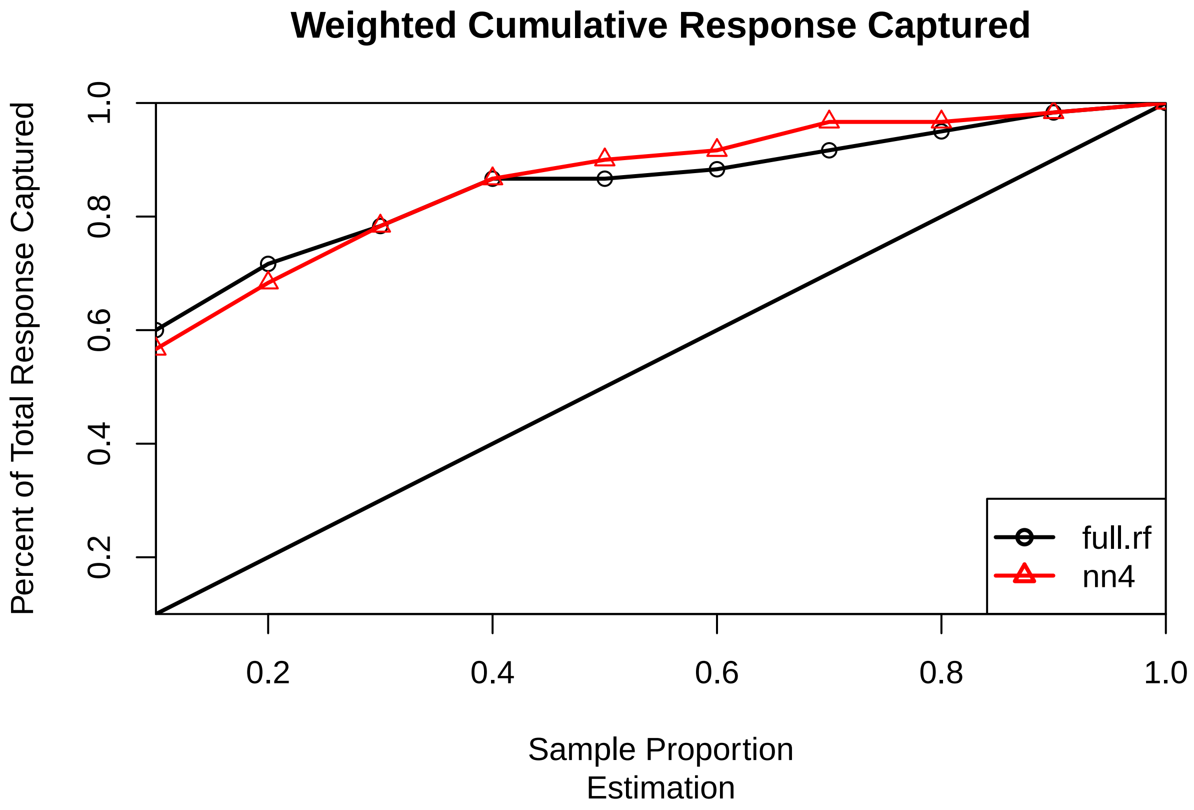


Lift Charts

```
lift.chart(modelList = c("full.rf", "nn6", "nn4"),
           data = train,
           targLevel = "Y", trueResp = 0.01,
           type = "cumulative", sub = "Estimation")
```



```
lift.chart(modelList = c("full.rf", "nn6", "nn4"),
           data = test,
           targLevel = "Y", trueResp = 0.01,
           type = "cumulative", sub = "Estimation")
```



Generating Predictions

```
data = read.csv("QK.csv", row.names = 'X')
data$Weeks3Meals <- NULL
data$Title <- NULL
#Number of days passed since last delivery
data$LastOrder = as.Date(data$LastOrder)
data$day_since_lastorder = as.numeric(as.Date("2018-03-05")-data$LastOrder)
data$LastOrder <- NULL
#Log
data$Log.DA_Income <- log(data$DA_Income)
#Changing NA to no discount
data$Disc <- fct_explicit_na(data$Disc, "NoDisc")
holdout = filter(data, data$Sample == "Holdout")
holdout$Log.DA_Income <- log(holdout$DA_Income)
holdout$nnet4.pred <- predict(nn4, holdout)
holdout$rf.pred <- predict(full.rf, holdout, type = "prob")[, 'Y']
holdout$rf2.pred <- predict(rf2, holdout, type = "prob")[, 'Y']
holdout$rf3.pred <- predict(rf3, holdout, type = "prob")[, 'Y']
holdout$ensemble <- rowMeans(holdout %>% dplyr::select(nnet4.pred, rf.pred, rf2.pred, rf3.pred))
holdout <- holdout %>% dplyr::select(custid, nnet4.pred, rf.pred, rf2.pred, rf3.pred)
colnames(holdout) <- c("custid", "score1", "score2", "score3", "score4")
write.csv(holdout, "ColorfulWRCAsst2_v4.csv")
```