APAN5420 — HW 10, Credit Card Transactions

Megan Wilder

8/3/18

Contents

---

1 Down-Sampling for the Majority Class

```
#split dataset into training and test sets
#set seed
set.seed(123)

# Sample into 3 sets. 60% train, 20% validation and 20% test
idx <-
sample(
seq(1, 3),
size = nrow(ccard),
replace = TRUE,
prob = c(.6, .2, .2)
)
train <- ccard[idx == 1,]
test <- ccard[idx == 2,]
val <- ccard[idx == 3,]

#check classes distribution
kable(prop.table(table(train$Class)))
```

Var1	Freq
0	0.9982625
1	0.0017375

```
kable(prop.table(table(test$Class)))
```

Var1	Freq
0	0.9983397
1	0.0016603

```
kable(prop.table(table(val$Class)))
```

Var1	Freq
0	0.998235
1	0.001765

```
#down-sampling, sample so that fraud represents about 10% of data set
#a typical range for resampling is to make fraud 5-20% of the training set.
#want to make it a significant amount of the training set but not
#amplyify the noise too much.
data_balanced_under <-
ovun.sample(
Class ~ .,
data = train,
method = "under",
p = 0.1,
seed = 1
)$data

#view table of class variable in rebalanced training set
kable(table(data_balanced_under$Class))
```

Var1	Freq
0	2661
1	297

```
#view classes distribution in rebalanced training set
kable(prop.table(table(data_balanced_under$Class)))
```

Var1	Freq
0	0.8995943
1	0.1004057

```
#Start H2O
h2o.init(nthreads = -1, max_mem_size = '8G')

# clean slate in case the cluster was already running
h2o.removeAll()
```

2 GBM Base Model, Without Tuning Hyperparameters

A gradient boosting machine computes a sequence of weak learners (typically very simple trees), where each successive tree is built for the prediction residuals of the preceding tree. It is an ensemble method, which combines several base models to produce one optimal predictive model. The combined estimator is usually better than any of the single base estimators as its bias is reduced.

```
# make h2o data.frame, loads into H2O service
train.hex <- as.h2o(data_balanced_under)
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
test.hex <- as.h2o(test)
```

```
##
  |
```

```
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
val.hex <- as.h2o(val)
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
# Response and predictors to use
resp <- "Class"
pred <- setdiff(names(train.hex), 'Class')

# Build a baseline gbm model without hyperparameter tuning
gbm <- h2o.gbm(x = pred,
y = resp,
training_frame = train.hex)
```

```
##
  |
  |                                                                      |   0%
  |
  |======                                                                |  10%
  |
  |=================================================================== |  98%
  |
  |======================================================================| 100%
```

```
gbm
```

```
## Model Details:
## ==============
##
## H2OBinomialModel: gbm
## Model ID:  GBM_model_R_1532977815003_96798
## Model Summary:
##   number_of_trees number_of_internal_trees model_size_in_bytes min_depth
## 1              50                       50               13359         5
##   max_depth mean_depth min_leaves max_leaves mean_leaves
## 1         5    5.00000          9         26    16.28000
##
##
## H2OBinomialMetrics: gbm
## ** Reported on training data. **
##
## MSE:  0.002473658
## RMSE:  0.04973588
## LogLoss:  0.01360717
## Mean Per-Class Error:  0.0001878993
## AUC:  0.9999962
## Gini:  0.9999924
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
```

```
##           0   1    Error     Rate
## 0      2660   1 0.000376  =1/2661
## 1         0 297 0.000000   =0/297
## Totals 2660 298 0.000338  =1/2958
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.244161 0.998319 115
## 2                       max f2  0.244161 0.999327 115
## 3                 max f0point5  0.265519 0.997963 111
## 4                 max accuracy  0.244161 0.999662 115
## 5                max precision  0.994057 1.000000   0
## 6                   max recall  0.244161 1.000000 115
## 7              max specificity  0.994057 1.000000   0
## 8             max absolute_mcc  0.244161 0.998133 115
## 9   max min_per_class_accuracy  0.244161 0.999624 115
## 10 max mean_per_class_accuracy  0.244161 0.999812 115
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<model>, valid=<T/
```

```
# Get the AUC on the validation set
h2o.auc(h2o.performance(gbm, newdata = val.hex)) #0.9875728
```

```
## [1] 0.9875728
```

3 Tune Hyperparameters with Grid Search

Hyperparameters:
Learning rate (shrinkage): A value between 0 and 1, corresponds to how quickly the error is corrected from each tree to the next. A lower learning rate is generally better, but will require more trees (and computational time). A large learning rate makes the system unable to settle down.
Learn rate annealing: hyperparameter to decay the learning rate. Speeds up convergence without sacrificing too much accuracy. Max Depth: The maximum allowed depth for the trees. Deeper trees take longer to train.
Sample Rate: Row sampling rate, can improve generalization and lead to lower validation and test set errors. Rule of thumb for large datasets is around 0.7 to 0.8 (sampling 70-80% of the data).
Column Sample Rate: Column sampling rate, can improve generalization and lead to lower validation and test set errors. Rule of thumb for large datasets is around 0.7 to 0.8 (sampling 70-80% of the data).
Number of trees: # of trees used
score_tree_interval = 10: Score every 10 trees to make early stopping reproducible.
max_runtime_secs=1200: Early stopping based on timeout. In this case no more than 1200 seconds.
stopping_rounds = 5,
stopping_tolerance = 1e-4,
stopping_metric = "AUC",
The above three hyperparameters control the early stopping when the AUC does not improve by at least 0.01% for 5 consecutive scoring events.

Grid Search: I used H2o's grid search to train and validate numerous models at once based on different hyper-parameter levels.

(Source: https://blog.h2o.ai/2016/06/h2o-gbm-tuning-tutorial-for-r/)

```
#create list of hyperparameters to tune
hyper_params <- list(
learn_rate = c(0.01, .05, 0.1),
max_depth = seq(2, 12, 2),
```

```
sample_rate = c(0.7, 0.8, 1.0),
col_sample_rate = c(0.7, 0.8, 1.0),
ntrees = seq(500, 2000, 500)
)

#Cartesian Grid Search
grid <- h2o.grid(
hyper_params = hyper_params,
search_criteria = list(strategy = "Cartesian"),
algorithm = "gbm",
grid_id = "gbm_grid",
x = pred,
y = resp,
training_frame = train.hex,
validation_frame = val.hex,
seed = 123,
learn_rate_annealing = .99,
max_runtime_secs = 1200,
#Early stopping based on timeout. In this case no more than 1200 seconds.
stopping_rounds = 5,
stopping_tolerance = 1e-4,
stopping_metric = "AUC",
#The above three hyperparameters control the early stopping when
#the AUC does not improve by at least 0.01% for 5 consecutive
#scoring events.
score_tree_interval = 10 #Score every 10 trees to make early stopping reproducible.
)
```

```
#view grid
grid
```

```
## H2O Grid Details
## ================
##
## Grid ID: gbm_grid
## Used hyper parameters:
##   -  col_sample_rate
##   -  learn_rate
##   -  max_depth
##   -  ntrees
##   -  sample_rate
## Number of models: 648
## Number of failed models: 0
##
## Hyper-Parameter Search Summary: ordered by increasing logloss
##   col_sample_rate learn_rate max_depth ntrees sample_rate
## 1             0.8        0.1        12    500         1.0
## 2             0.8        0.1        12   1500         1.0
## 3             0.8        0.1        12   2000         1.0
## 4             0.8        0.1        12   1000         1.0
## 5             0.7        0.1        12   2000         1.0
##            model_ids              logloss
## 1 gbm_grid_model_484 0.007396540859659651
## 2 gbm_grid_model_592 0.007396540859659651
```

```
## 3 gbm_grid_model_646 0.007396540859659651
## 4 gbm_grid_model_538 0.007396540859659651
## 5 gbm_grid_model_645   0.0078326703570576
##
## ---
##     col_sample_rate learn_rate max_depth ntrees sample_rate
## 643             1.0       0.01         6   1500         0.7
## 644             1.0       0.01         6    500         0.7
## 645             0.7       0.01         4   2000         1.0
## 646             0.7       0.01         4   1000         1.0
## 647             0.7       0.01         4    500         1.0
## 648             0.7       0.01         4   1500         1.0
##              model_ids             logloss
## 643 gbm_grid_model_128 0.06391086719527837
## 644  gbm_grid_model_20 0.06391086719527837
## 645 gbm_grid_model_603 0.06455329298183389
## 646 gbm_grid_model_495 0.06455329298183389
## 647 gbm_grid_model_441 0.06455329298183389
## 648 gbm_grid_model_549 0.06455329298183389
```

```
## sort the grid models by decreasing AUC
sortedGrid <-
h2o.getGrid("gbm_grid", sort_by = "auc", decreasing = TRUE)
print(sortedGrid)
```

```
## H2O Grid Details
## ================
##
## Grid ID: gbm_grid
## Used hyper parameters:
##   -  col_sample_rate
##   -  learn_rate
##   -  max_depth
##   -  ntrees
##   -  sample_rate
## Number of models: 648
## Number of failed models: 0
##
## Hyper-Parameter Search Summary: ordered by decreasing auc
##   col_sample_rate learn_rate max_depth ntrees sample_rate
## 1             0.7        0.1        10    500         0.7
## 2             0.7        0.1        10   2000         0.7
## 3             0.7        0.1        10   1000         0.7
## 4             0.7        0.1        10   1500         0.7
## 5             0.7        0.1         8   1000         1.0
##            model_ids                auc
## 1  gbm_grid_model_42  0.990980073200488
## 2 gbm_grid_model_204  0.990980073200488
## 3  gbm_grid_model_96  0.990980073200488
## 4 gbm_grid_model_150  0.990980073200488
## 5 gbm_grid_model_519 0.9906016054599784
##
## ---
##     col_sample_rate learn_rate max_depth ntrees sample_rate
## 643             1.0       0.01         2    500         1.0
```

```
## 644             1.0       0.01         2   2000         1.0
## 645             0.7       0.01         2   1500         1.0
## 646             0.7       0.01         2    500         1.0
## 647             0.7       0.01         2   2000         1.0
## 648             0.7       0.01         2   1000         1.0
##              model_ids                auc
## 643 gbm_grid_model_434 0.9390906519086939
## 644 gbm_grid_model_596 0.9390906519086939
## 645 gbm_grid_model_540 0.9376048057711689
## 646 gbm_grid_model_432 0.9376048057711689
## 647 gbm_grid_model_594 0.9376048057711689
## 648 gbm_grid_model_486 0.9376048057711689
```

```
#print AUC for 10 best models
for (i in 1:10) {
topModels <- h2o.getModel(sortedGrid@model_ids[[i]])
print(h2o.auc(h2o.performance(topModels, valid = TRUE)))
} #best model had AUC of 0.9909801 on validation set
```

```
## [1] 0.9909801
## [1] 0.9909801
## [1] 0.9909801
## [1] 0.9909801
## [1] 0.9906016
## [1] 0.9906016
## [1] 0.9906016
## [1] 0.9906016
## [1] 0.9904544
## [1] 0.9904544
```

```
#name model with highest AUC best model
best_model <-
h2o.getModel(sortedGrid@model_ids[[1]])
#better than my base model, which had an AUC of 0.9875728 on
#its validation set

#best model's parameters
best_model@parameters
```

```
## $model_id
## [1] "gbm_grid_model_42"
##
## $training_frame
## [1] "data_balanced_under"
##
## $validation_frame
## [1] "val"
##
## $score_tree_interval
## [1] 10
##
## $ntrees
## [1] 500
##
## $max_depth
## [1] 10
```

```
##
## $stopping_rounds
## [1] 5
##
## $stopping_metric
## [1] "AUC"
##
## $stopping_tolerance
## [1] 1e-04
##
## $max_runtime_secs
## [1] 1200
##
## $seed
## [1] 123
##
## $learn_rate_annealing
## [1] 0.99
##
## $distribution
## [1] "bernoulli"
##
## $sample_rate
## [1] 0.7
##
## $col_sample_rate
## [1] 0.7
##
## $x
##  [1] "Time"   "V1"     "V2"     "V3"     "V4"     "V5"     "V6"
##  [8] "V7"     "V8"     "V9"     "V10"    "V11"    "V12"    "V13"
## [15] "V14"    "V15"    "V16"    "V17"    "V18"    "V19"    "V20"
## [22] "V21"    "V22"    "V23"    "V24"    "V25"    "V26"    "V27"
## [29] "V28"    "Amount"
##
## $y
## [1] "Class"
```

```
#view best model
summary(best_model)
```

```
## Model Details:
## ==============
##
## H2OBinomialModel: gbm
## Model Key:  gbm_grid_model_42
## Model Summary:
##   number_of_trees number_of_internal_trees model_size_in_bytes min_depth
## 1             120                      120               68622        10
##   max_depth mean_depth min_leaves max_leaves mean_leaves
## 1        10   10.00000         19         67    40.25834
##
## H2OBinomialMetrics: gbm
## ** Reported on training data. **
##
```

```
## MSE:  0.000380636
## RMSE:  0.0195099
## LogLoss:  0.003441163
## Mean Per-Class Error:  0
## AUC:  1
## Gini:  1
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
##           0   1    Error     Rate
## 0      2661   0 0.000000  =0/2661
## 1         0 297 0.000000   =0/297
## Totals 2661 297 0.000000  =0/2958
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.529397 1.000000 128
## 2                       max f2  0.529397 1.000000 128
## 3                 max f0point5  0.529397 1.000000 128
## 4                 max accuracy  0.529397 1.000000 128
## 5                max precision  0.999302 1.000000   0
## 6                   max recall  0.529397 1.000000 128
## 7              max specificity  0.999302 1.000000   0
## 8             max absolute_mcc  0.529397 1.000000 128
## 9   max min_per_class_accuracy  0.529397 1.000000 128
## 10 max mean_per_class_accuracy  0.529397 1.000000 128
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<model>, valid=<T/
## H2OBinomialMetrics: gbm
## ** Reported on validation data. **
##
## MSE:  0.001931355
## RMSE:  0.04394718
## LogLoss:  0.0090302
## Mean Per-Class Error:  0.1050707
## AUC:  0.9909801
## Gini:  0.9819601
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
##            0  1    Error       Rate
## 0      56549  8 0.000141   =8/56557
## 1         21 79 0.210000    =21/100
## Totals 56570 87 0.000512  =29/56657
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.992619 0.844920   4
## 2                       max f2  0.940160 0.836614  17
## 3                 max f0point5  0.994975 0.883028   3
## 4                 max accuracy  0.992619 0.999488   4
## 5                max precision  0.994975 0.916667   3
## 6                   max recall  0.000225 1.000000 396
## 7              max specificity  0.999002 0.999876   0
## 8             max absolute_mcc  0.992619 0.846720   4
## 9   max min_per_class_accuracy  0.015349 0.960000 332
```

```
## 10 max mean_per_class_accuracy  0.114689 0.964934 223
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<model>, valid=<T/
##
##
## Scoring History:
##              timestamp          duration number_of_trees training_rmse
## 1  2018-08-02 16:30:21  1 min 27.491 sec               0       0.30054
## 2  2018-08-02 16:30:21  1 min 27.565 sec              10       0.13901
## 3  2018-08-02 16:30:21  1 min 27.718 sec              20       0.09361
## 4  2018-08-02 16:30:21  1 min 27.873 sec              30       0.07442
## 5  2018-08-02 16:30:22  1 min 28.041 sec              40       0.06248
## 6  2018-08-02 16:30:22  1 min 28.195 sec              50       0.05253
## 7  2018-08-02 16:30:22  1 min 28.356 sec              60       0.04142
## 8  2018-08-02 16:30:22  1 min 28.519 sec              70       0.03469
## 9  2018-08-02 16:30:22  1 min 28.675 sec              80       0.03017
## 10 2018-08-02 16:30:22  1 min 28.836 sec              90       0.02622
## 11 2018-08-02 16:30:22  1 min 28.993 sec             100       0.02399
## 12 2018-08-02 16:30:23  1 min 29.150 sec             110       0.02159
## 13 2018-08-02 16:30:23  1 min 29.305 sec             120       0.01951
##    training_logloss training_auc training_lift
## 1           0.32597      0.50000       1.00000
## 2           0.09682      0.99911       9.95960
## 3           0.05105      0.99965       9.95960
## 4           0.03179      0.99984       9.95960
## 5           0.02161      0.99997       9.95960
## 6           0.01552      0.99999       9.95960
## 7           0.01066      1.00000       9.95960
## 8           0.00804      1.00000       9.95960
## 9           0.00651      1.00000       9.95960
## 10          0.00532      1.00000       9.95960
## 11          0.00456      1.00000       9.95960
## 12          0.00392      1.00000       9.95960
## 13          0.00344      1.00000       9.95960
##    training_classification_error validation_rmse validation_logloss
## 1                        0.89959         0.10720            0.10968
## 2                        0.00913         0.06171            0.05158
## 3                        0.00609         0.05151            0.03031
## 4                        0.00406         0.04999            0.02159
## 5                        0.00169         0.04730            0.01669
## 6                        0.00101         0.04702            0.01427
## 7                        0.00000         0.04548            0.01203
## 8                        0.00000         0.04512            0.01091
## 9                        0.00000         0.04504            0.01034
## 10                       0.00000         0.04508            0.00994
## 11                       0.00000         0.04401            0.00938
## 12                       0.00000         0.04387            0.00913
## 13                       0.00000         0.04395            0.00903
##    validation_auc validation_lift validation_classification_error
## 1         0.50000         1.00000                         0.99823
## 2         0.98437        87.93326                         0.00062
## 3         0.99055        87.93326                         0.00056
## 4         0.99132        88.93250                         0.00055
## 5         0.99110        90.93099                         0.00056
```

```
## 6         0.99128        92.92947                          0.00055
## 7         0.99143        91.93023                          0.00053
## 8         0.99109        91.93023                          0.00053
## 9         0.99126        92.92947                          0.00051
## 10        0.99073        92.92947                          0.00051
## 11        0.99076        92.92947                          0.00051
## 12        0.99092        92.92947                          0.00051
## 13        0.99098        92.92947                          0.00051
##
## Variable Importances: (Extract with `h2o.varimp`)
## =================================================
##
## Variable Importances:
##   variable relative_importance scaled_importance percentage
## 1      V14          676.443115          1.000000   0.540722
## 2      V10          313.391876          0.463294   0.250513
## 3      V17           75.471916          0.111572   0.060329
## 4      V12           27.852541          0.041175   0.022264
## 5       V4           21.005356          0.031053   0.016791
##
## ---
##    variable relative_importance scaled_importance percentage
## 25      V18            2.372698          0.003508   0.001897
## 26       V2            1.690066          0.002498   0.001351
## 27      V25            1.514799          0.002239   0.001211
## 28      V21            1.463194          0.002163   0.001170
## 29      V27            1.176335          0.001739   0.000940
## 30      V28            0.782543          0.001157   0.000626
```

```
#get the actual number of trees
ntrees <- best_model@parameters[["ntrees"]]
ntrees #500
```

```
## [1] 500
```

```
#get the actual max depth
mdepth <- best_model@parameters[["max_depth"]]
mdepth #10
```

```
## [1] 10
```

```
#Validation set used to select the best model
#Evaluate the model performance on test set to get honest estimate of model performance
best_model_perf <- h2o.performance(model = best_model,
newdata = test.hex)

#model performance metrics on test set
best_model_perf
```

```
## H2OBinomialMetrics: gbm
##
## MSE:  0.00233282
## RMSE:  0.04829927
## LogLoss:  0.0112636
## Mean Per-Class Error:  0.1054382
## AUC:  0.9631125
## Gini:  0.9262249
```

```
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
##            0  1    Error       Rate
## 0      57103 20 0.000350  =20/57123
## 1         20 75 0.210526     =20/95
## Totals 57123 95 0.000699  =40/57218
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.978064 0.789474   9
## 2                       max f2  0.978064 0.789474   9
## 3                 max f0point5  0.995736 0.797267   4
## 4                 max accuracy  0.982575 0.999301   8
## 5                max precision  0.998659 0.814286   1
## 6                   max recall  0.000180 1.000000 398
## 7              max specificity  0.999042 0.999772   0
## 8             max absolute_mcc  0.978064 0.789124   9
## 9   max min_per_class_accuracy  0.005659 0.905263 362
## 10 max mean_per_class_accuracy  0.028180 0.927348 310
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<model>, valid=<T/
```

```
#MSE
h2o.mse(best_model_perf) #0.00233282, not really relevant for classification problems
```

```
## [1] 0.00233282
```

```
#RMSE
h2o.rmse(best_model_perf) #0.04829927, not really relevant for classification problems
```

```
## [1] 0.04829927
```

```
#Log Loss
h2o.logloss(best_model_perf) #0.0112636
```

```
## [1] 0.0112636
```

```
#AUC
h2o.auc(best_model_perf)  #0.9631125, slighly less than the AUC on the validation set
```

```
## [1] 0.9631125
```

```
#Gini
h2o.giniCoef(best_model_perf) #0.9262249
```

```
## [1] 0.9262249
```

```
#best model performance metrics at all thresholds
test.scores <- best_model_perf@metrics$thresholds_and_metric_scores

#find best threshold that maximizes F1
best.thresh <- test.scores$threshold[which.max(test.scores$f1)]
best.thresh #0.9780643
```

```
## [1] 0.9780643
```

```
#create dataframe with performance metrics of model on test data at
#threshold that maximizes F1
metrics <- data_frame(
```

```
Precision = h2o.precision(best_model_perf, best.thresh),
Recall = h2o.recall(best_model_perf, best.thresh),
F1 = h2o.F1(best_model_perf, best.thresh),
AUC = h2o.auc(best_model_perf),
LogLoss = h2o.logloss(best_model_perf),
Gini = h2o.giniCoef(best_model_perf),
Accuracy = h2o.accuracy(best_model_perf, best.thresh),
Mean_Accuracy = h2o.mean_per_class_accuracy(best_model_perf, best.thresh)
)

#view metrics
kable(metrics) %>%
kable_styling(bootstrap_options = "striped", full_width = F)
```

Precision	Recall	F1	AUC	LogLoss	Gini	Accuracy	Mean_Accuracy
0.7894737	0.7894737	0.7894737	0.9631125	0.0112636	0.9262249	0.9993009	0.8945618

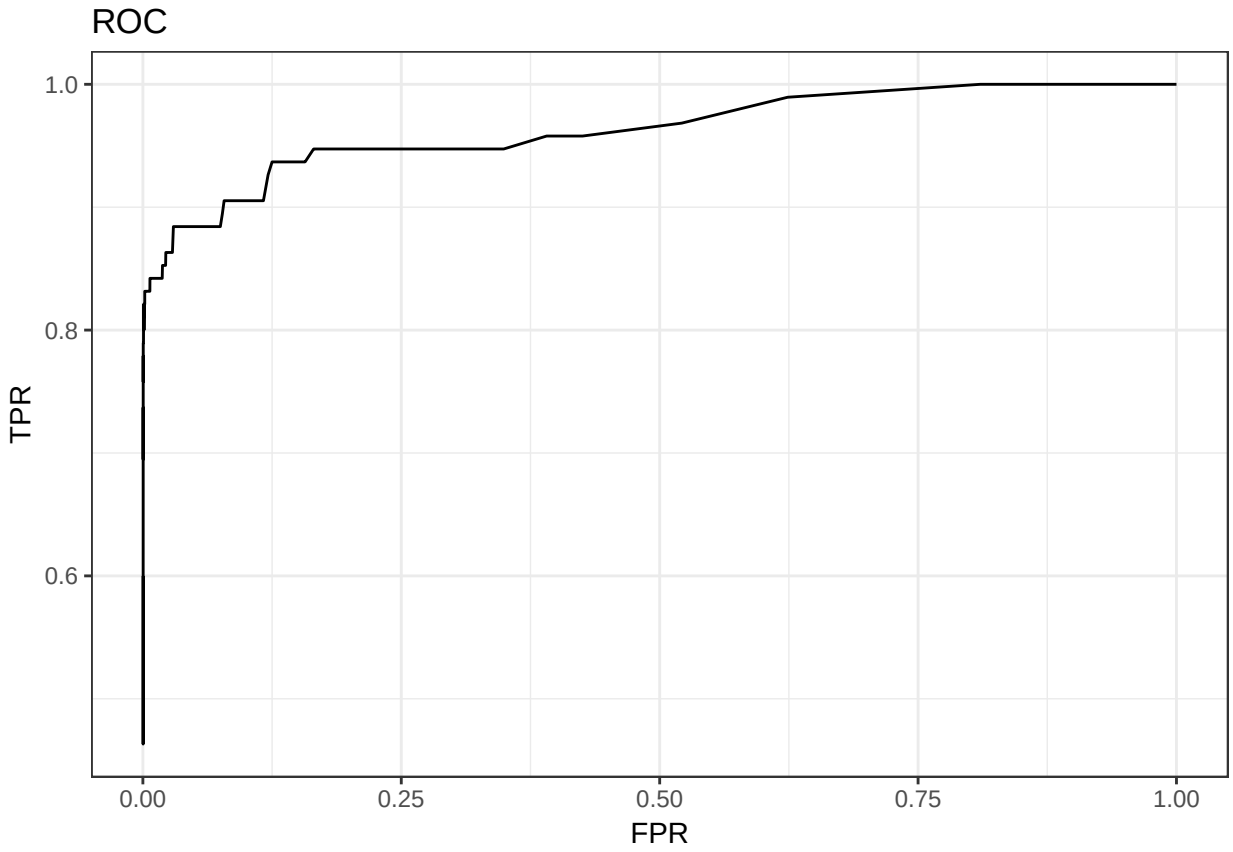
```
#overall it appears that my model performed well
```

4 Plot ROC, Precision-recall

```
#ROC
tpr = as.data.frame(h2o.tpr(best_model_perf))
fpr = as.data.frame(h2o.fpr(best_model_perf))
ROC_out <- merge(tpr, fpr, by = 'threshold')
head(ROC_out)
```

```
##      threshold       tpr       fpr
## 1 0.0001542087 1.0000000 1.0000000
## 2 0.0001798985 1.0000000 0.8102341
## 3 0.0002120161 0.9894737 0.6238643
## 4 0.0002545644 0.9684211 0.5214187
## 5 0.0002993347 0.9578947 0.4254153
## 6 0.0003408222 0.9578947 0.3906307
```

```
#Plot ROC
ggplot(ROC_out, aes(x = fpr, y = tpr)) +
theme_bw() +
geom_line() +
ggtitle("ROC") + ylab("TPR") + xlab("FPR")
```




```
#ROC curves plot the tradeoff between recall and false positive rates

#Precision Recall
#Evaluate the predictive performance of model based on precision and recall
head(h2o.F1(best_model_perf))
```

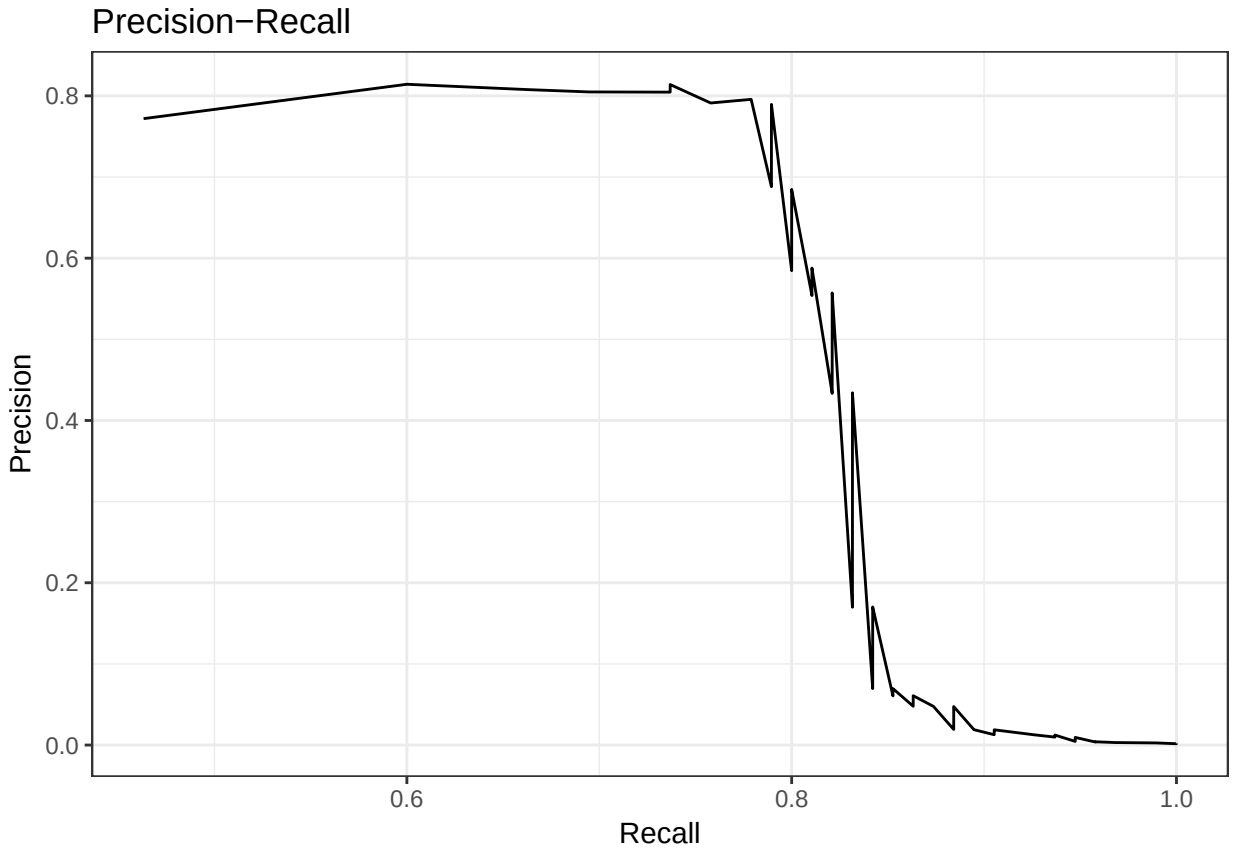
```
##   threshold        f1
## 1 0.9990416 0.5789474
## 2 0.9986595 0.6909091
## 3 0.9977966 0.7283237
## 4 0.9968699 0.7457627
## 5 0.9957357 0.7734807
## 6 0.9917144 0.7692308
```

```
precision = as.data.frame(h2o.precision(best_model_perf))
recall = as.data.frame(h2o.recall(best_model_perf))
PR_out <- merge(precision, recall, by = 'threshold')
head(PR_out)
```

```
##      threshold   precision       tpr
## 1 0.0001542087 0.001660317 1.0000000
## 2 0.0001798985 0.002048385 1.0000000
## 3 0.0002120161 0.002630769 0.9894737
## 4 0.0002545644 0.003079292 0.9684211
## 5 0.0002993347 0.003730731 0.9578947
## 6 0.0003408222 0.004061593 0.9578947
```

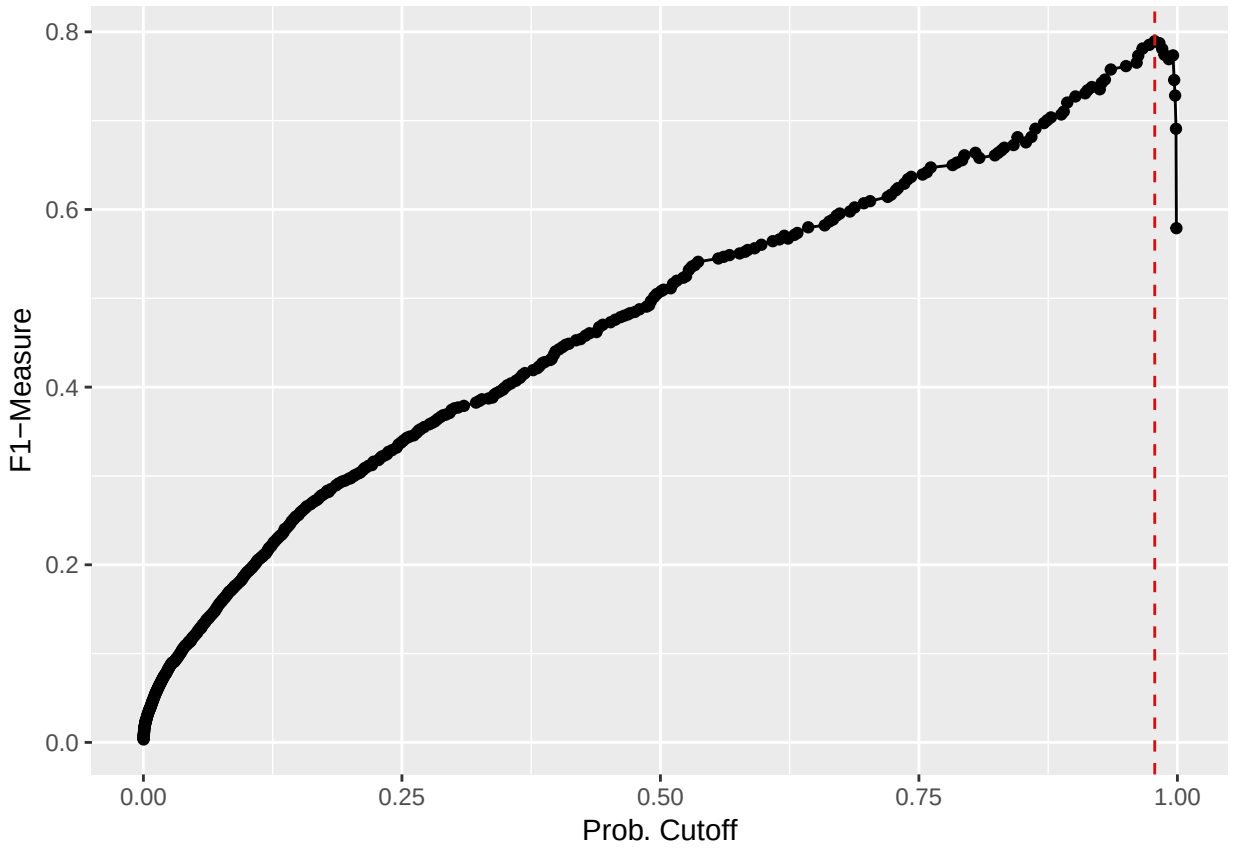
```
#Plot Precision - Recall
ggplot(PR_out, aes(x = tpr, y = precision)) +
```

```
theme_bw() +
geom_line() +
ggtitle("Precision-Recall") + ylab("Precision") + xlab("Recall")
```




```
#Precision-recall curves shows the tradeoff between precision and recall
#for different thresholds. Useful measure of prediction success when
#modeling rare events (classes very imbalanced).
#High precision relates to a low false positive rate,
#and high recall relates to a low false negative rate. High scores
#for both show that the classifier is returning accurate results
#(high precision), as well as returning a majority of all
#positive results (high recall).
#(source: http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html)

#plot threshold that maximizes F1
ggplot(test.scores, aes(x = threshold, y = f1)) +
geom_line() +
geom_point() +
geom_vline(xintercept = best.thresh,
linetype = "dashed",
color = "red") +
labs(x = "Prob. Cutoff", y = "F1-Measure")
```

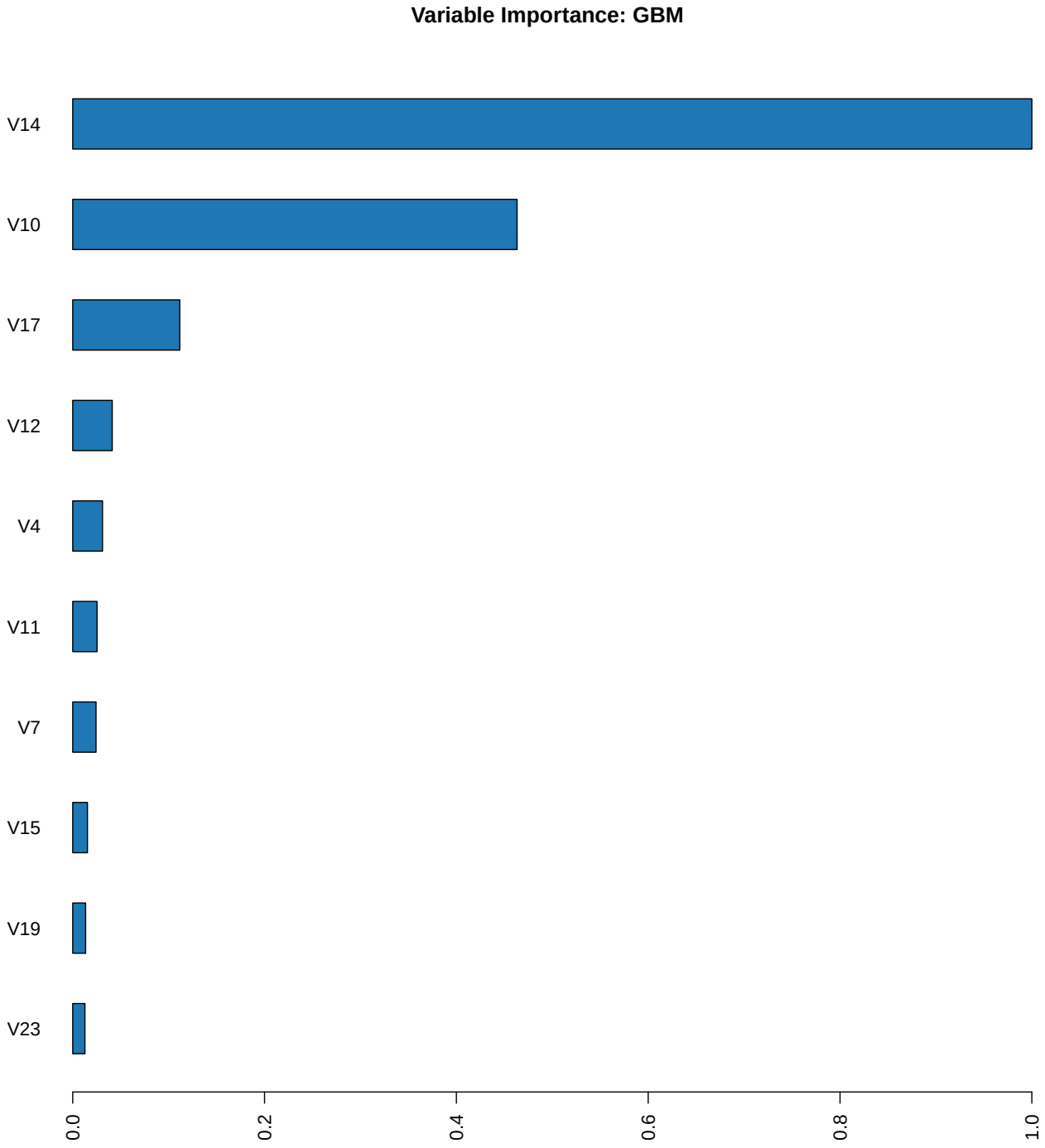


5 Variable Importance & Partial Dependence Plot

```
#variable importance
h2o.varimp(best_model)
```

```
## Variable Importances:
##    variable relative_importance scaled_importance percentage
## 1       V14          676.443115          1.000000   0.540722
## 2       V10          313.391876          0.463294   0.250513
## 3       V17           75.471916          0.111572   0.060329
## 4       V12           27.852541          0.041175   0.022264
## 5        V4           21.005356          0.031053   0.016791
##
## ---
##    variable relative_importance scaled_importance percentage
## 25      V18            2.372698          0.003508   0.001897
## 26       V2            1.690066          0.002498   0.001351
## 27      V25            1.514799          0.002239   0.001211
## 28      V21            1.463194          0.002163   0.001170
## 29      V27            1.176335          0.001739   0.000940
## 30      V28            0.782543          0.001157   0.000626
```

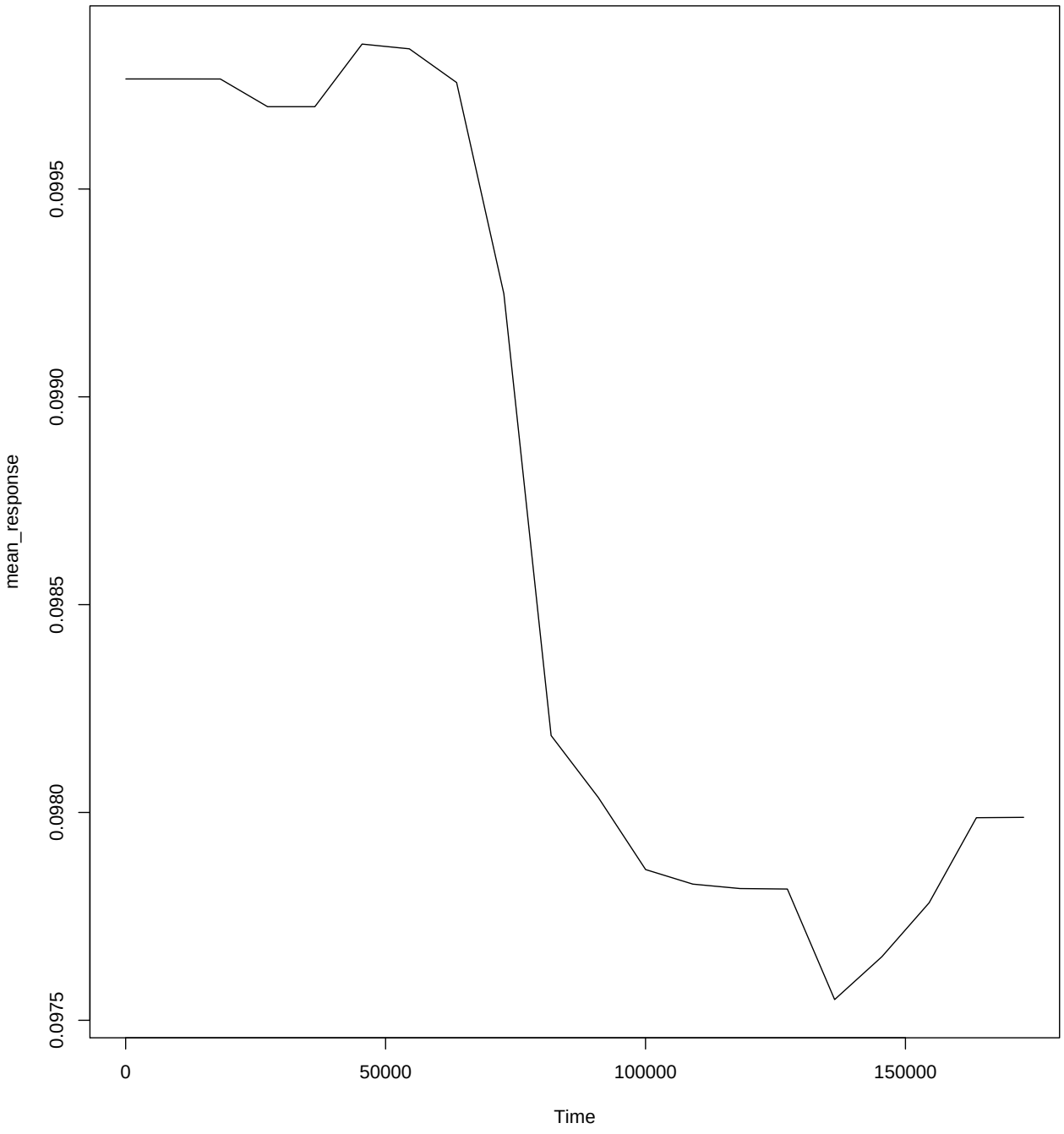
```
#plot variable importance
h2o.varimp_plot(best_model)
```

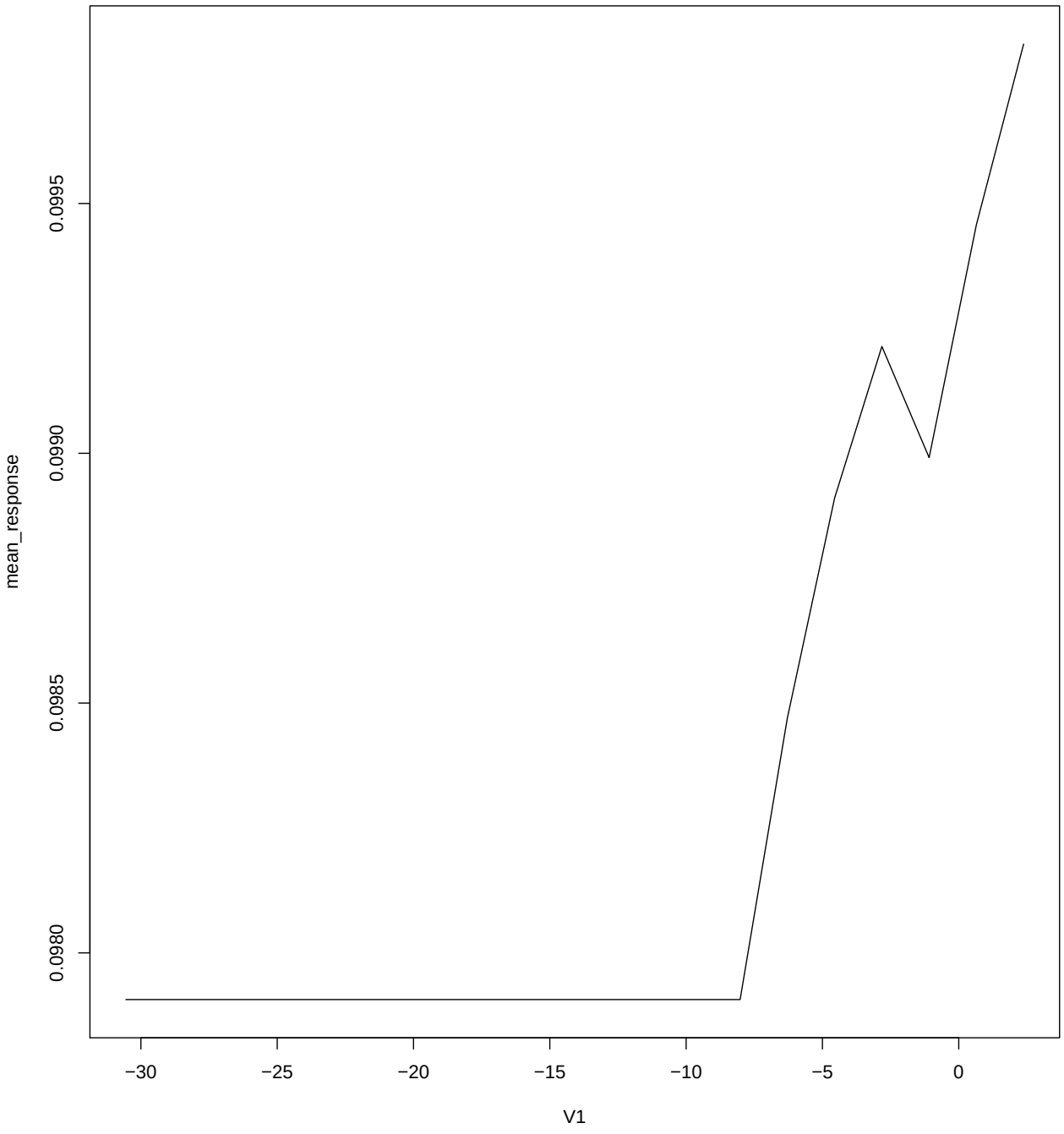


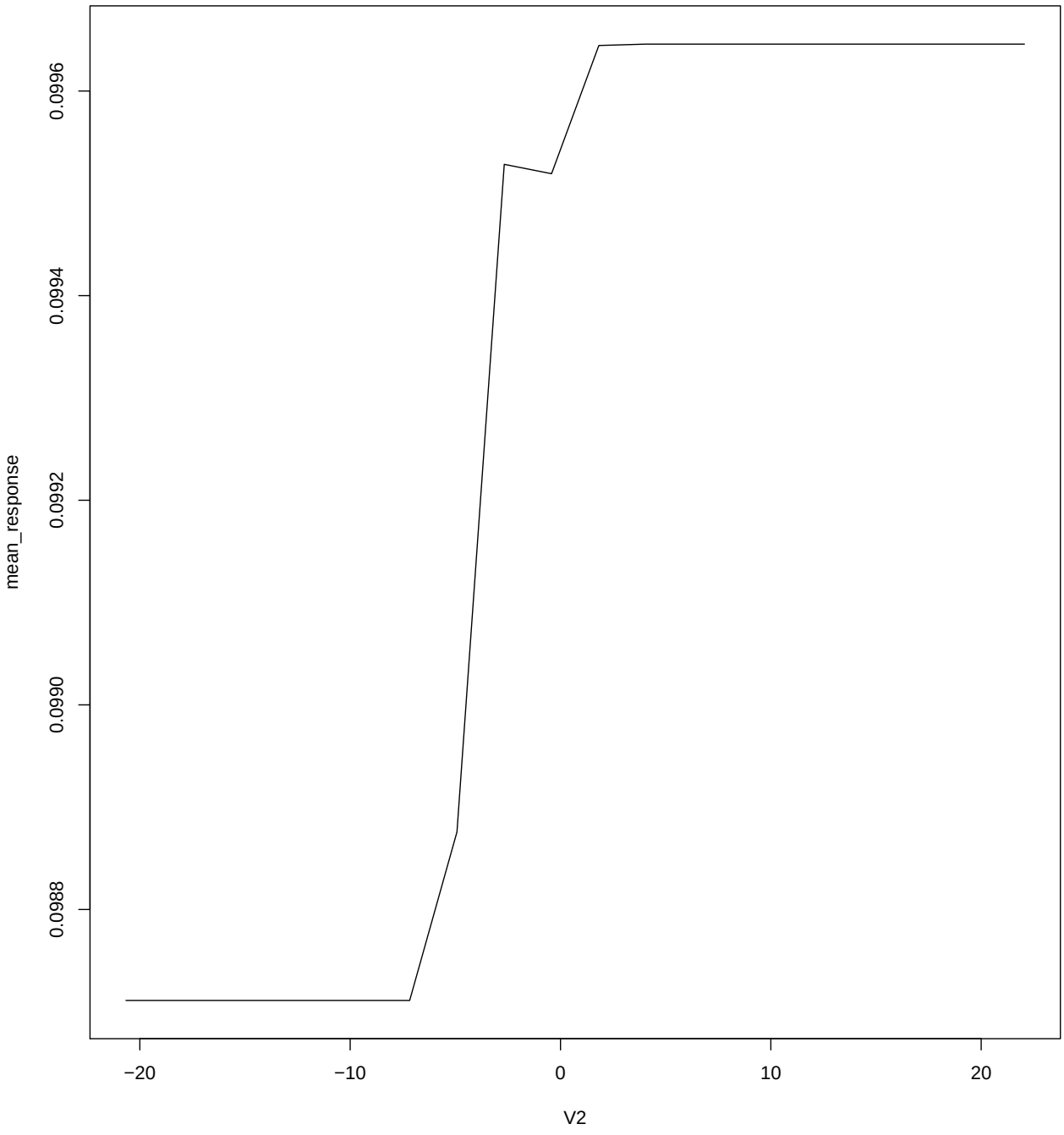

```
#partial dependence
h2o.partialPlot(
object = best_model,
data = train.hex,
plot = TRUE,
plot_stddev = FALSE
)
```

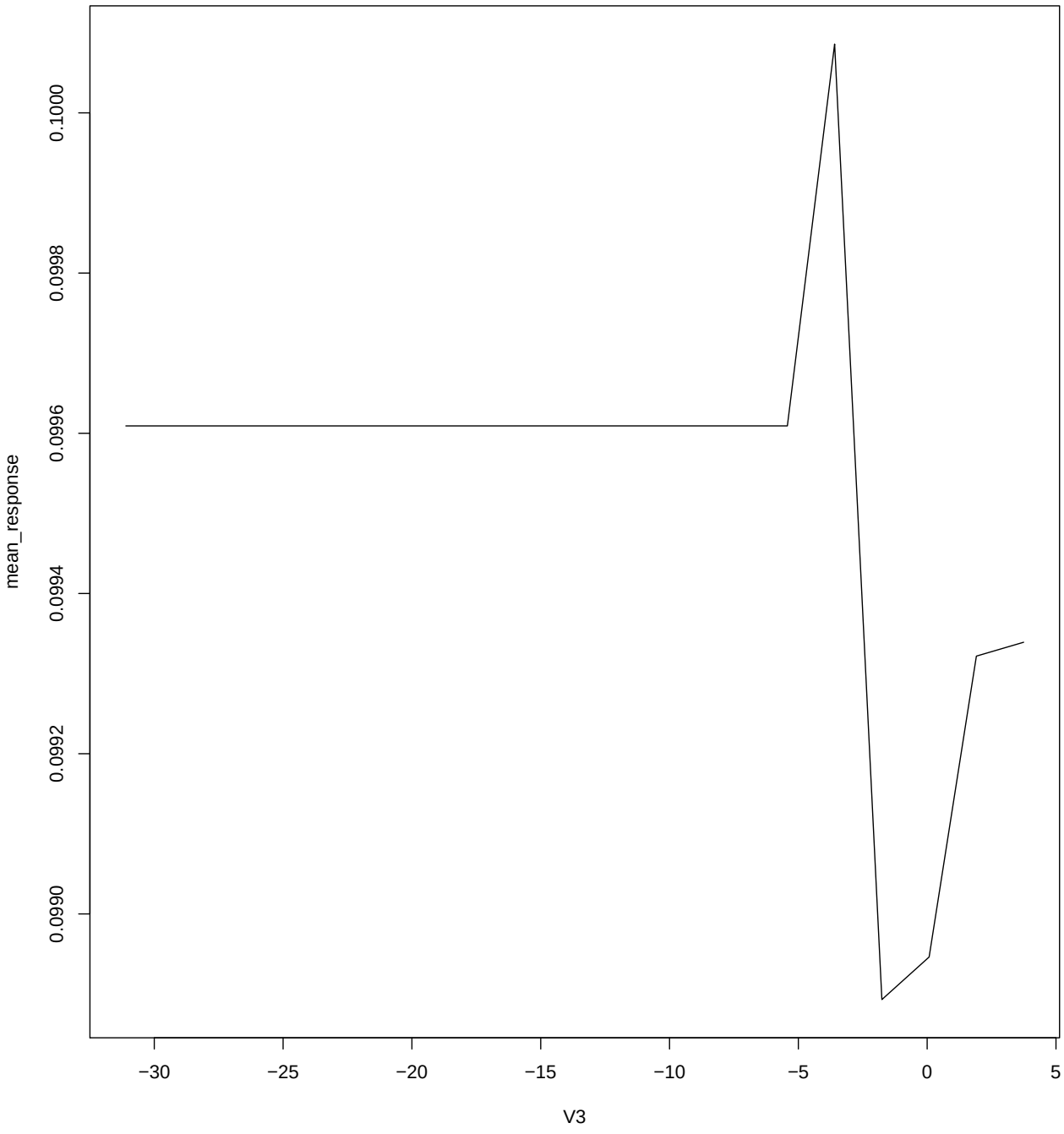
```
##
  |
  |                                                                      |   0%
```

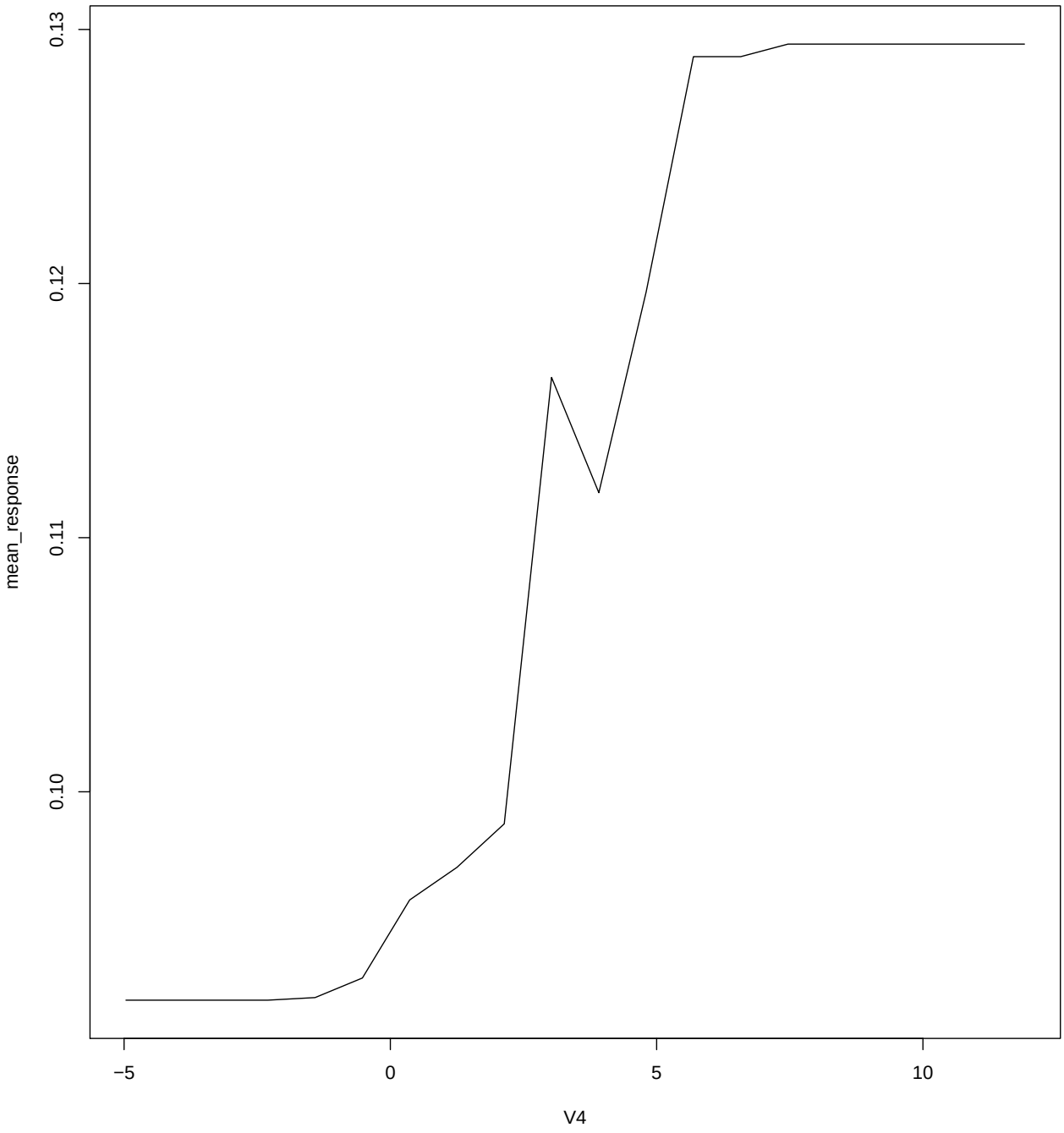
```
|
|==                                                                |   3%
|
|====                                                              |   7%
|
|=========                                                         |  13%
|
|===========                                                       |  17%
|
|==============                                                    |  20%
|
|===============                                                   |  23%
|
|=================                                                 |  27%
|
|====================                                              |  30%
|
|========================                                          |  37%
|
|===========================                                       |  40%
|
|==============================                                    |  47%
|
|================================                                  |  50%
|
|==================================                                |  57%
|
|======================================                            |  60%
|
|=======================================                           |  63%
|
|=============================================                     |  70%
|
|==============================================                    |  73%
|
|==================================================                |  80%
|
|====================================================              |  83%
|
|========================================================          |  90%
|
|============================================================      |  93%
|
|==============================================================    |  97%
|
|==================================================================| 100%
```

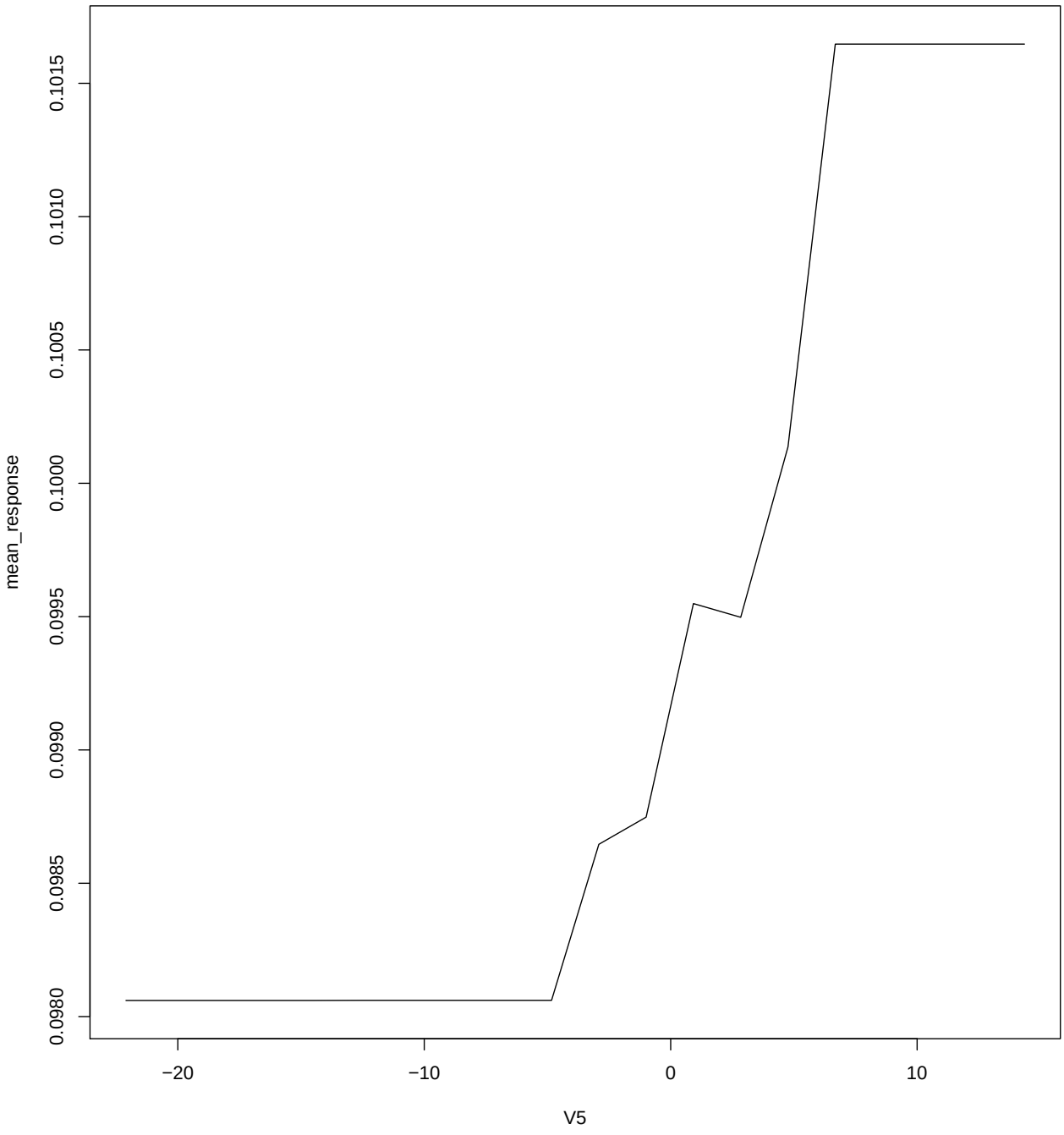

mean_response
0.0995
0.0990
0.0985
0.0980
0.0975
0
50000
100000
150000
Time

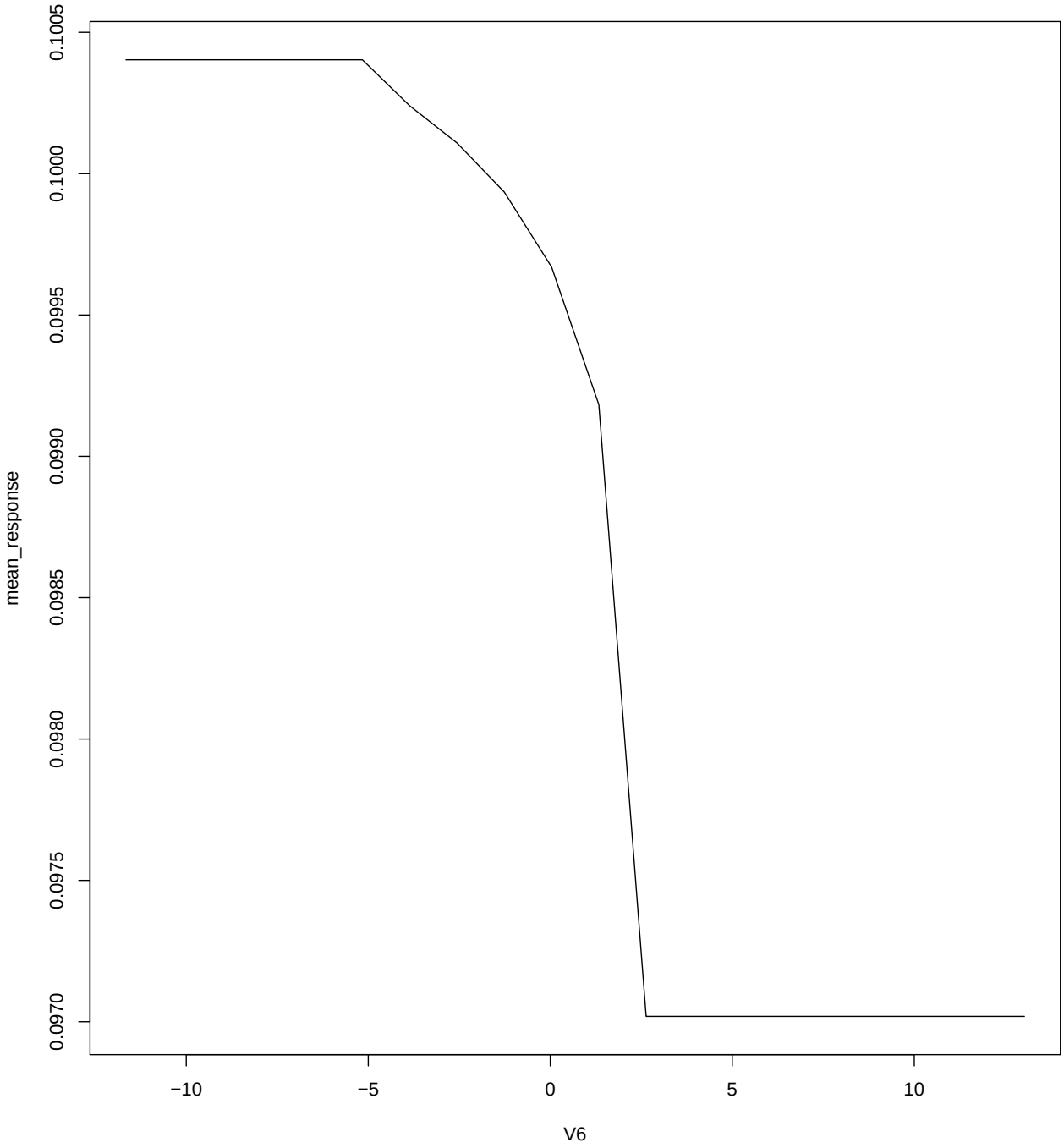

mean_response
0.0980
0.0985
0.0990
0.0995
−30
−25
−20
−15
−10
−5
0
V1

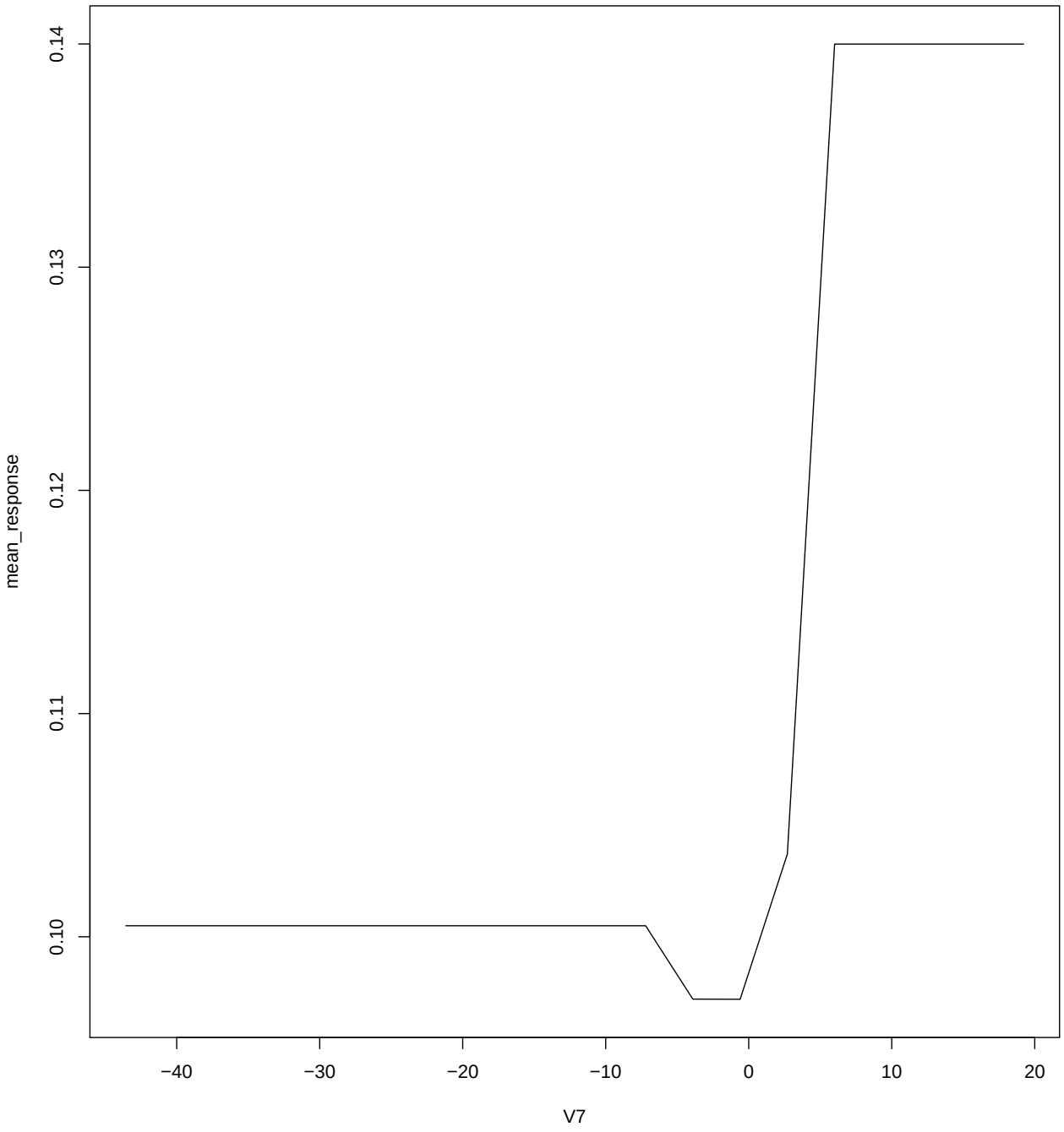

mean_response
0.0988
0.0990
0.0992
0.0994
0.0996
−20
−10
0
10
20
V2

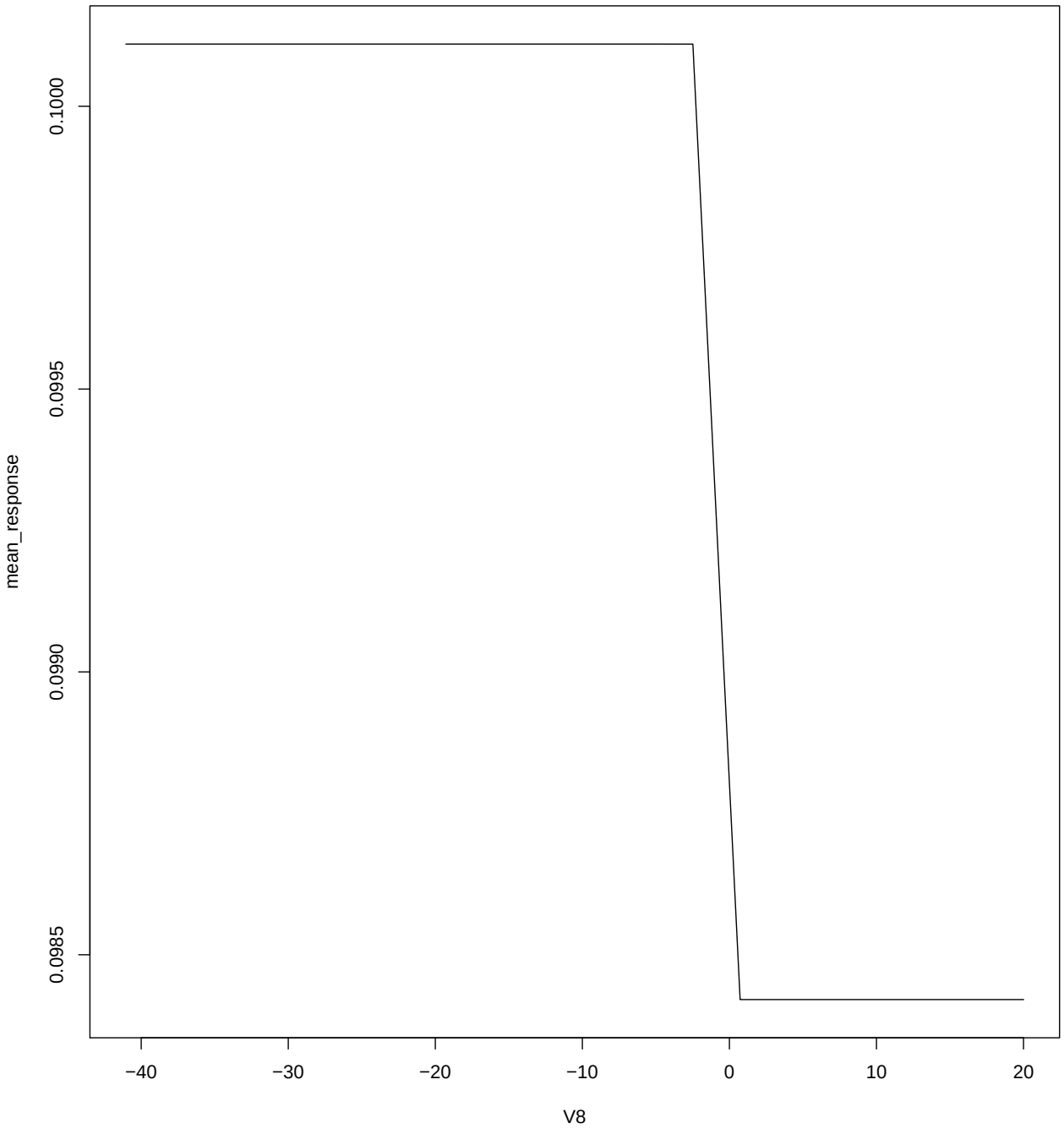

mean_response
0.1000
0.0998
0.0996
0.0994
0.0992
0.0990
−30
−25
−20
−15
−10
−5
0
5
V3

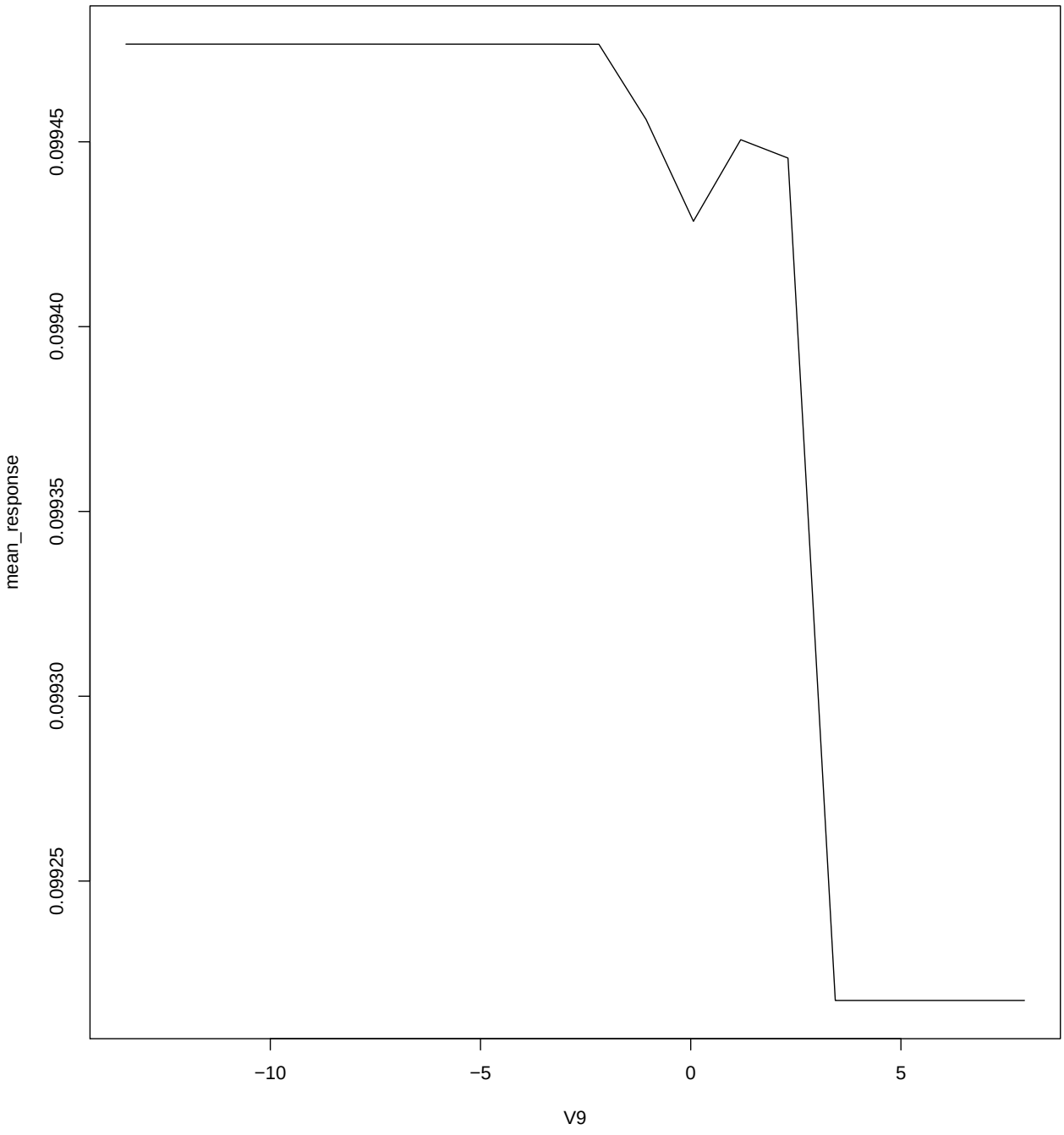

mean_response
0.10
0.11
0.12
0.13
−5
0
5
10
V4

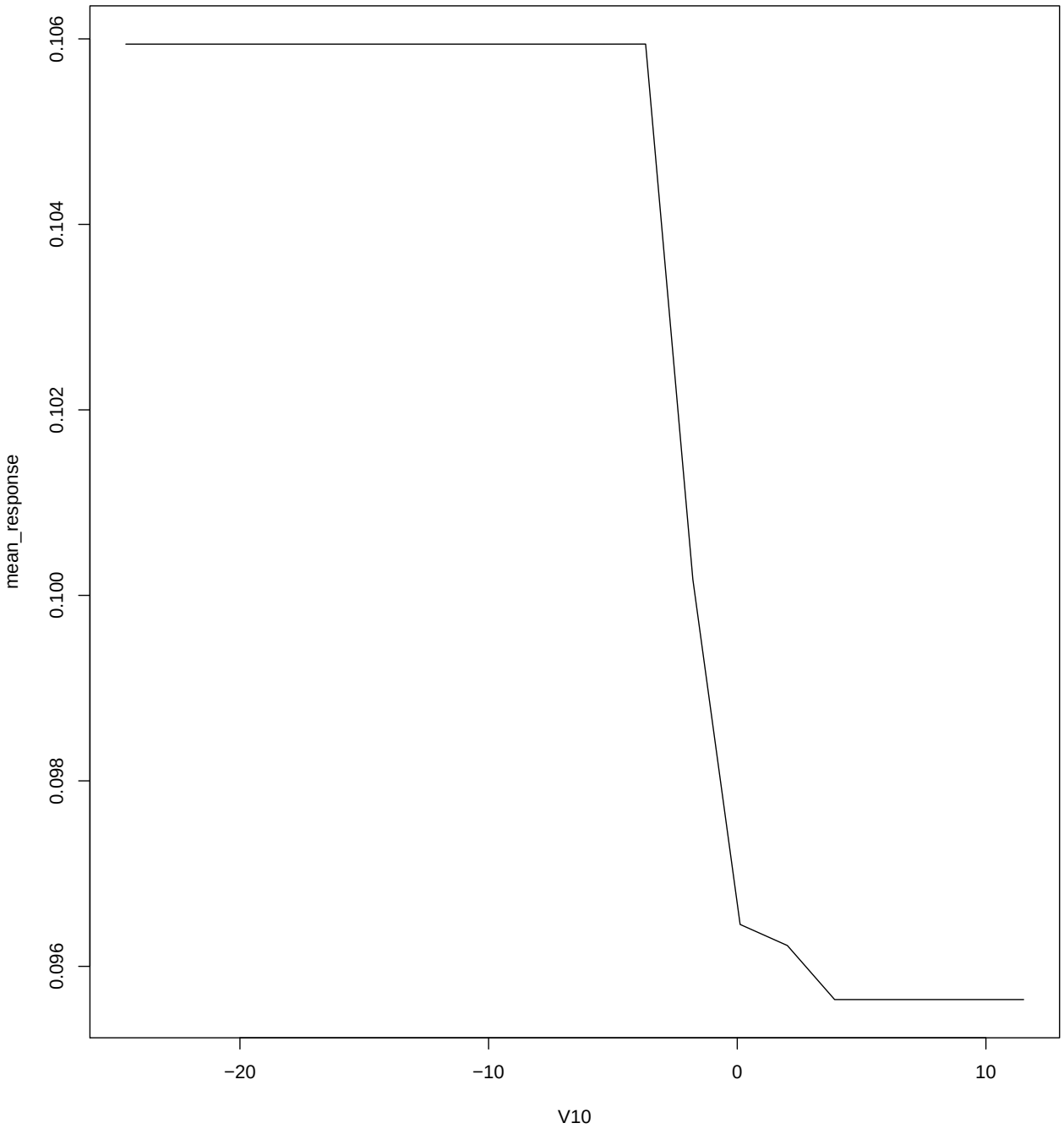

mean_response
0.0980
0.0985
0.0990
0.0995
0.1000
0.1005
0.1010
0.1015
−20
−10
0
10
V5

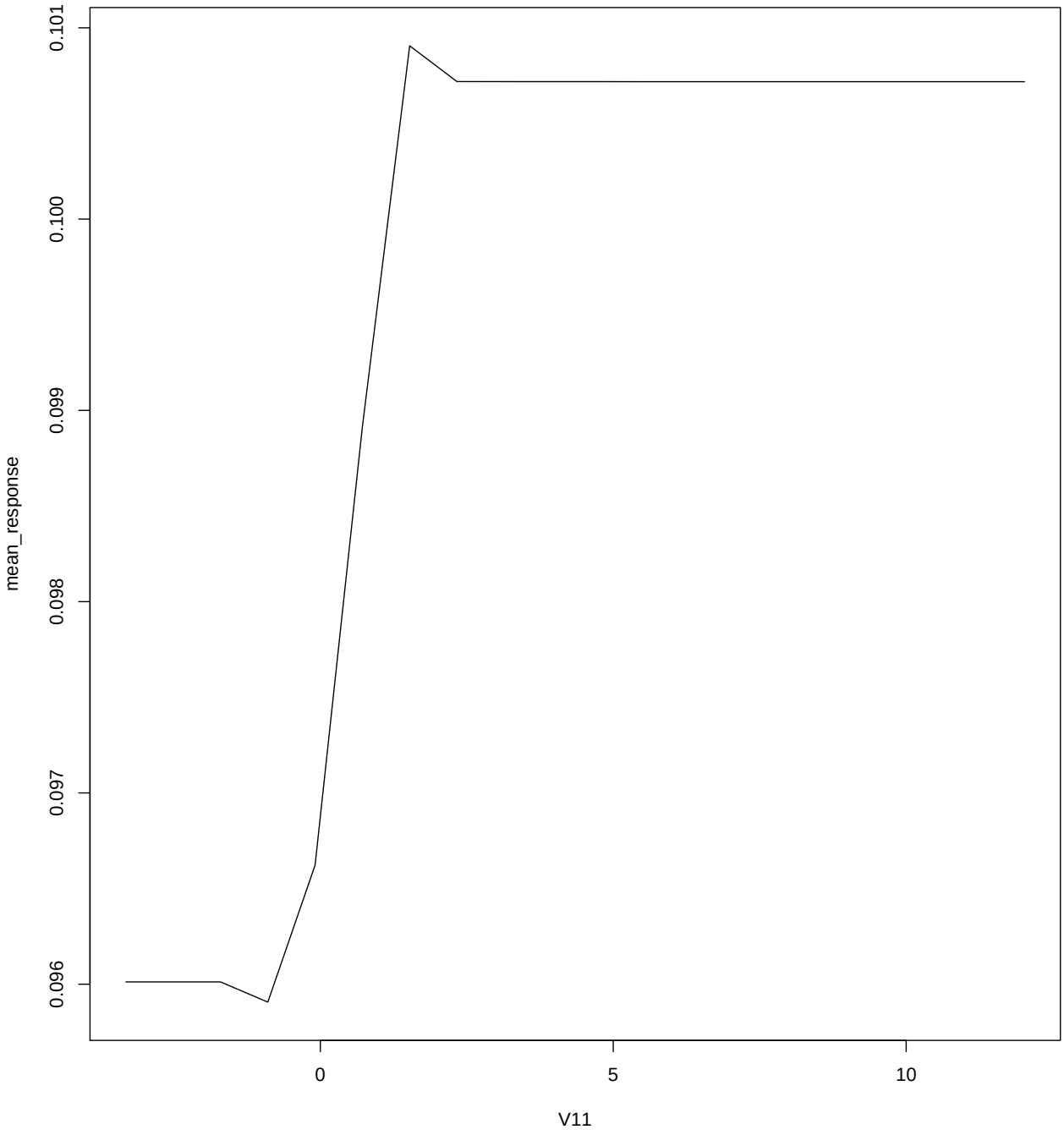

mean_response
0.1005
0.1000
0.0995
0.0990
0.0985
0.0980
0.0975
0.0970
−10
−5
0
5
10
V6

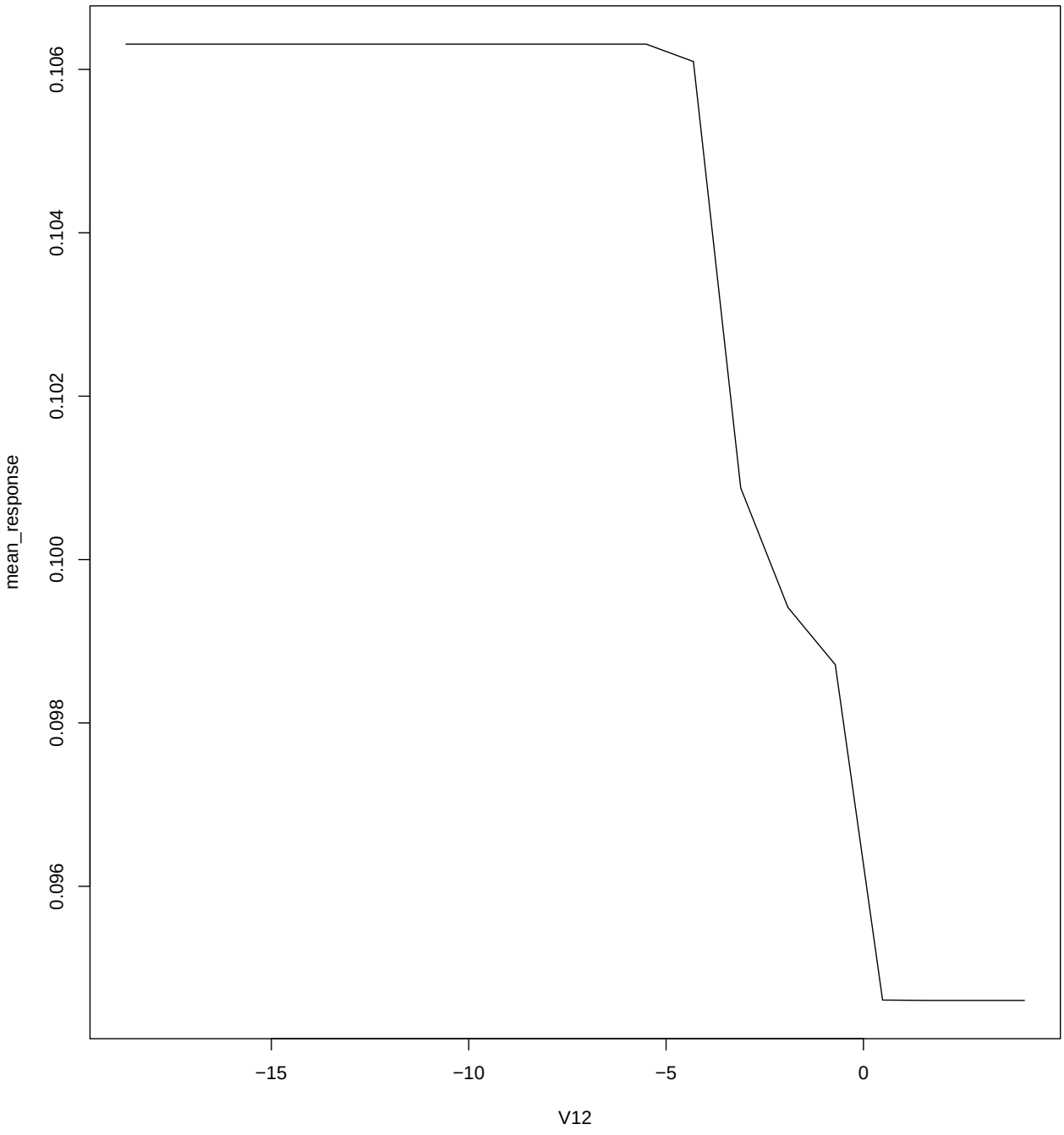

mean_response
0.10
0.11
0.12
0.13
0.14
−40
−30
−20
−10
0
10
20
V7

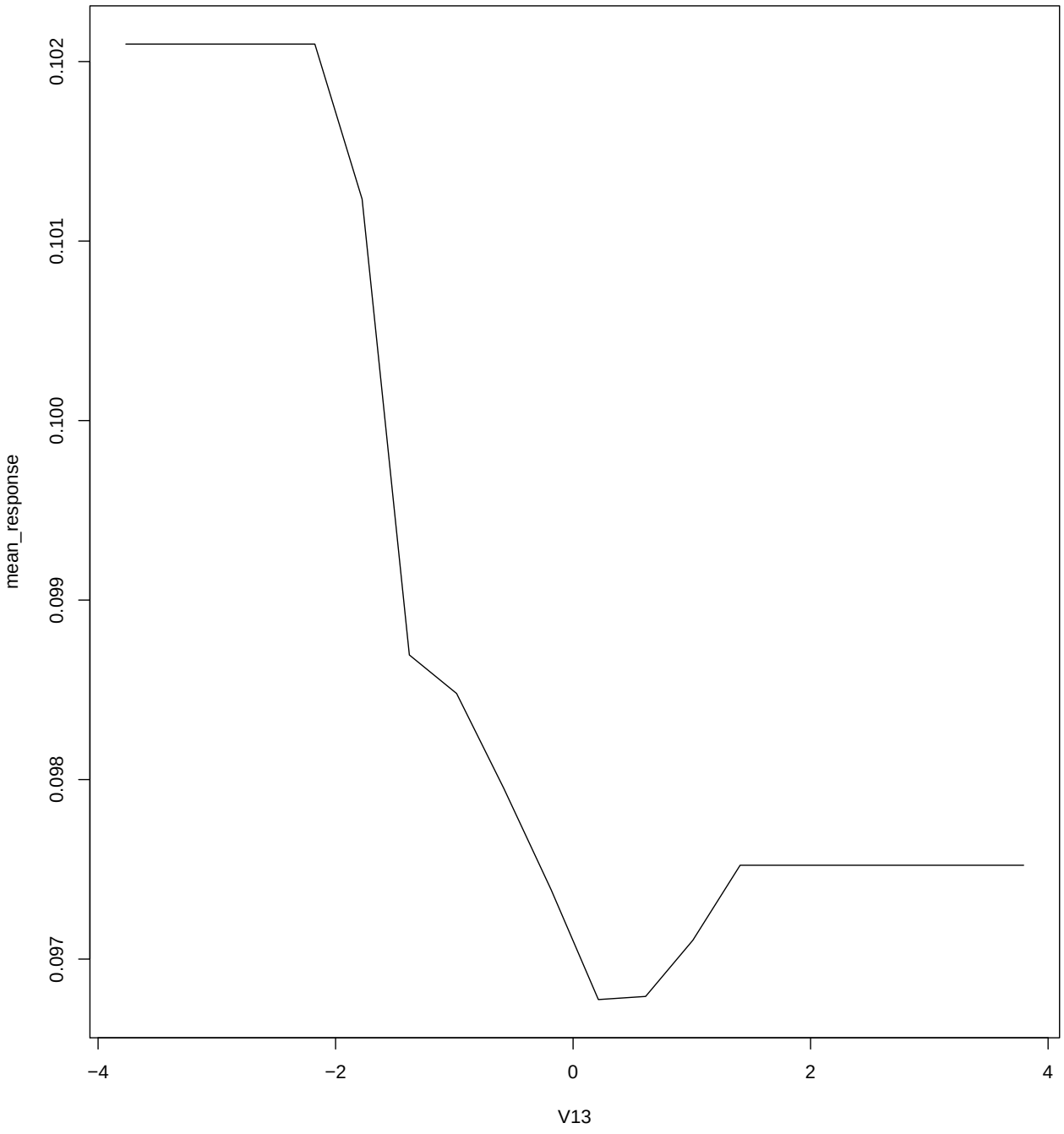

mean_response
0.1000
0.0995
0.0990
0.0985
−40
−30
−20
−10
0
10
20
V8

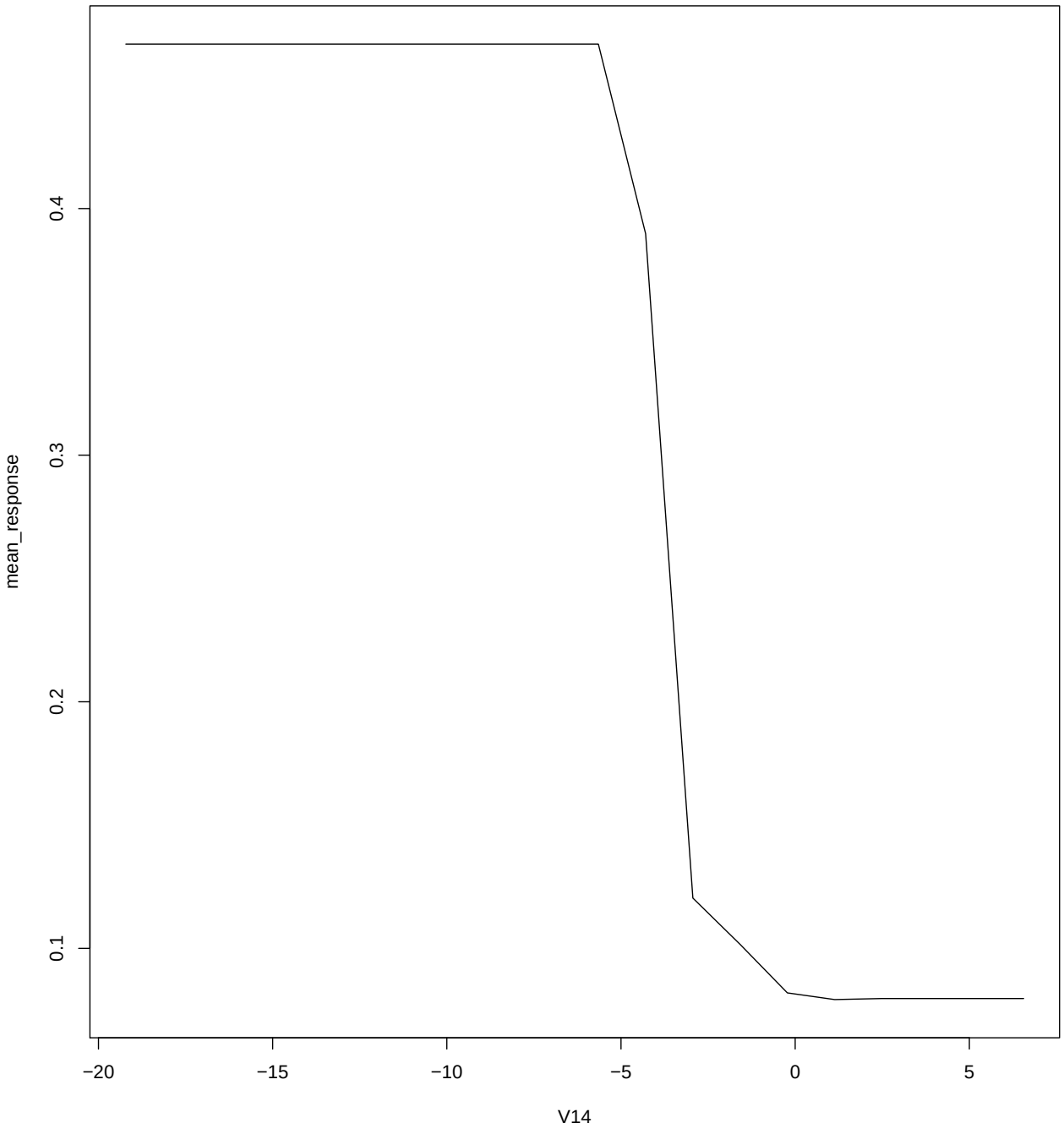

mean_response
0.09945
0.09940
0.09935
0.09930
0.09925
−10
−5
0
5
V9

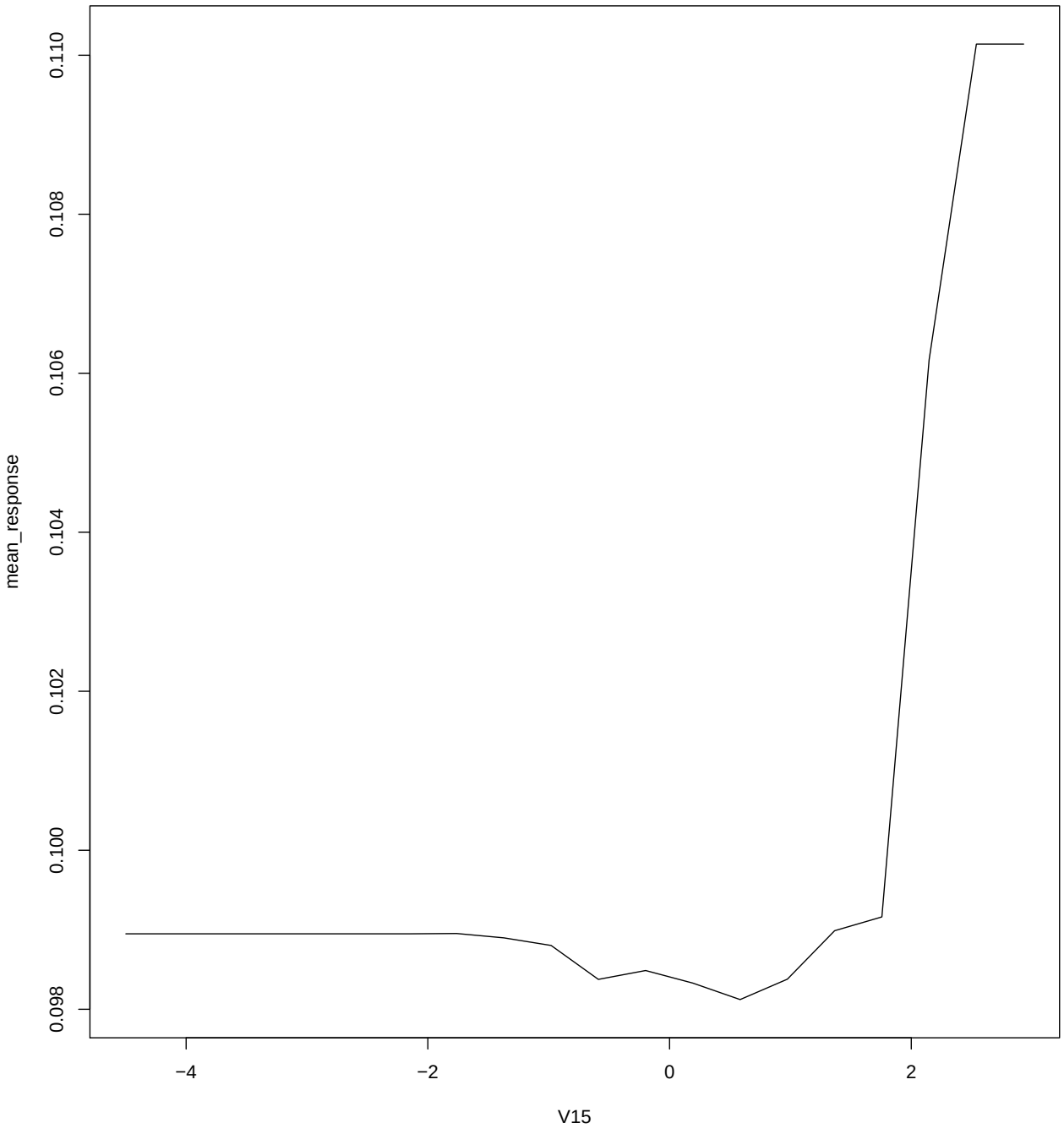

mean_response
0.096
0.098
0.100
0.102
0.104
0.106
−20
−10
0
10
V10

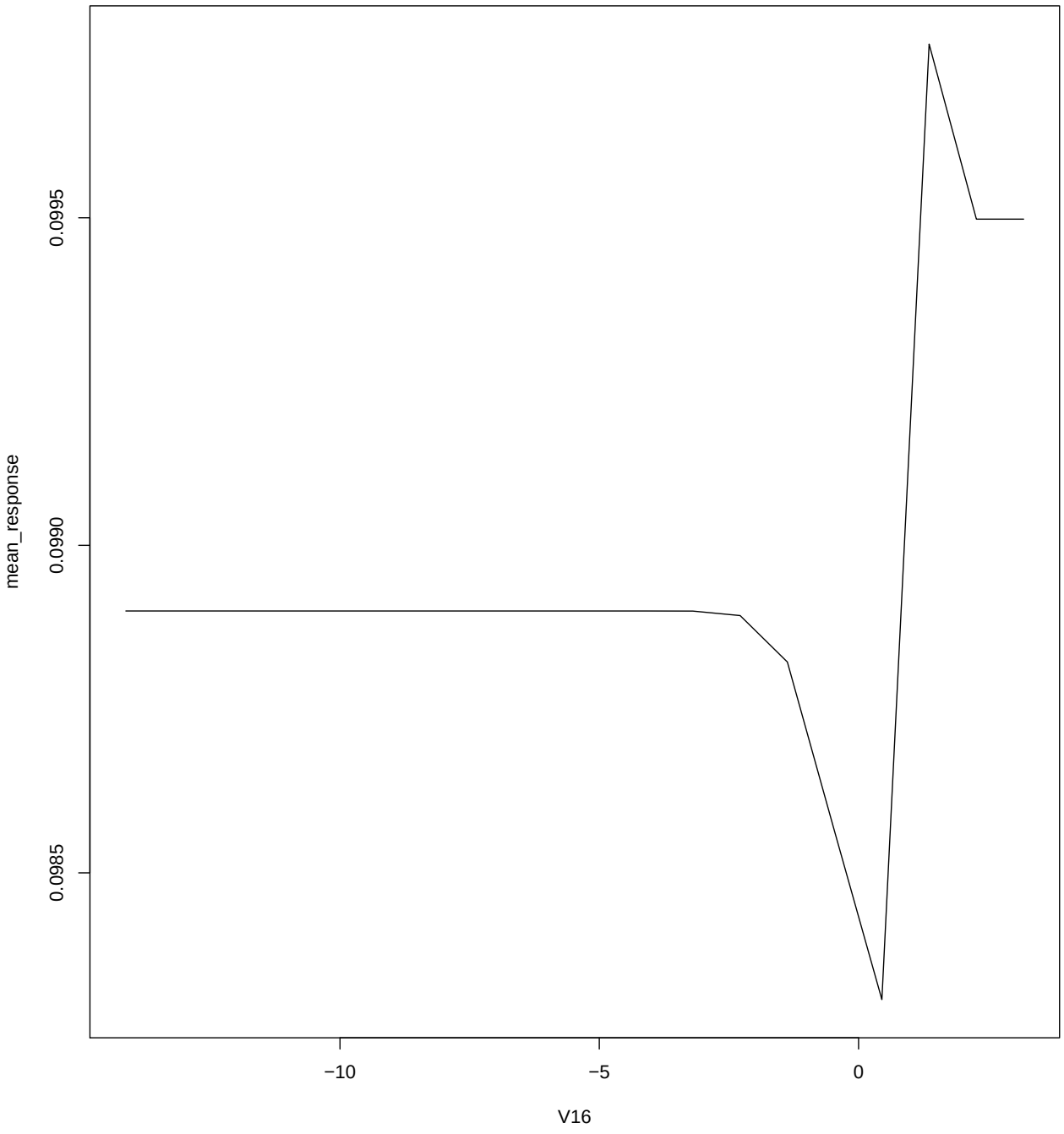

mean_response
0.096
0.097
0.098
0.099
0.100
0.101
0
5
10
V11

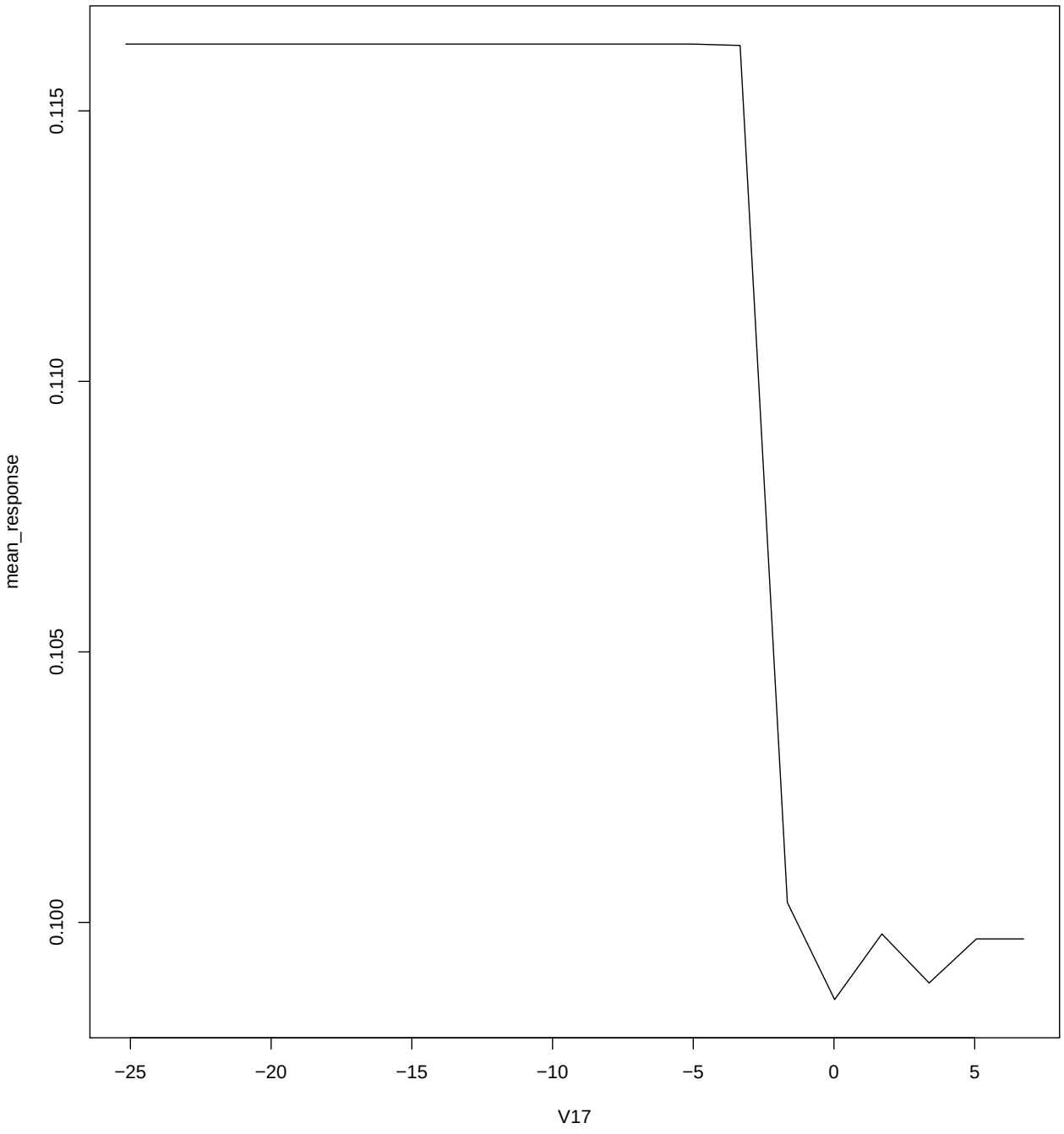

mean_response
0.096
0.098
0.100
0.102
0.104
0.106
−15
−10
−5
0
V12

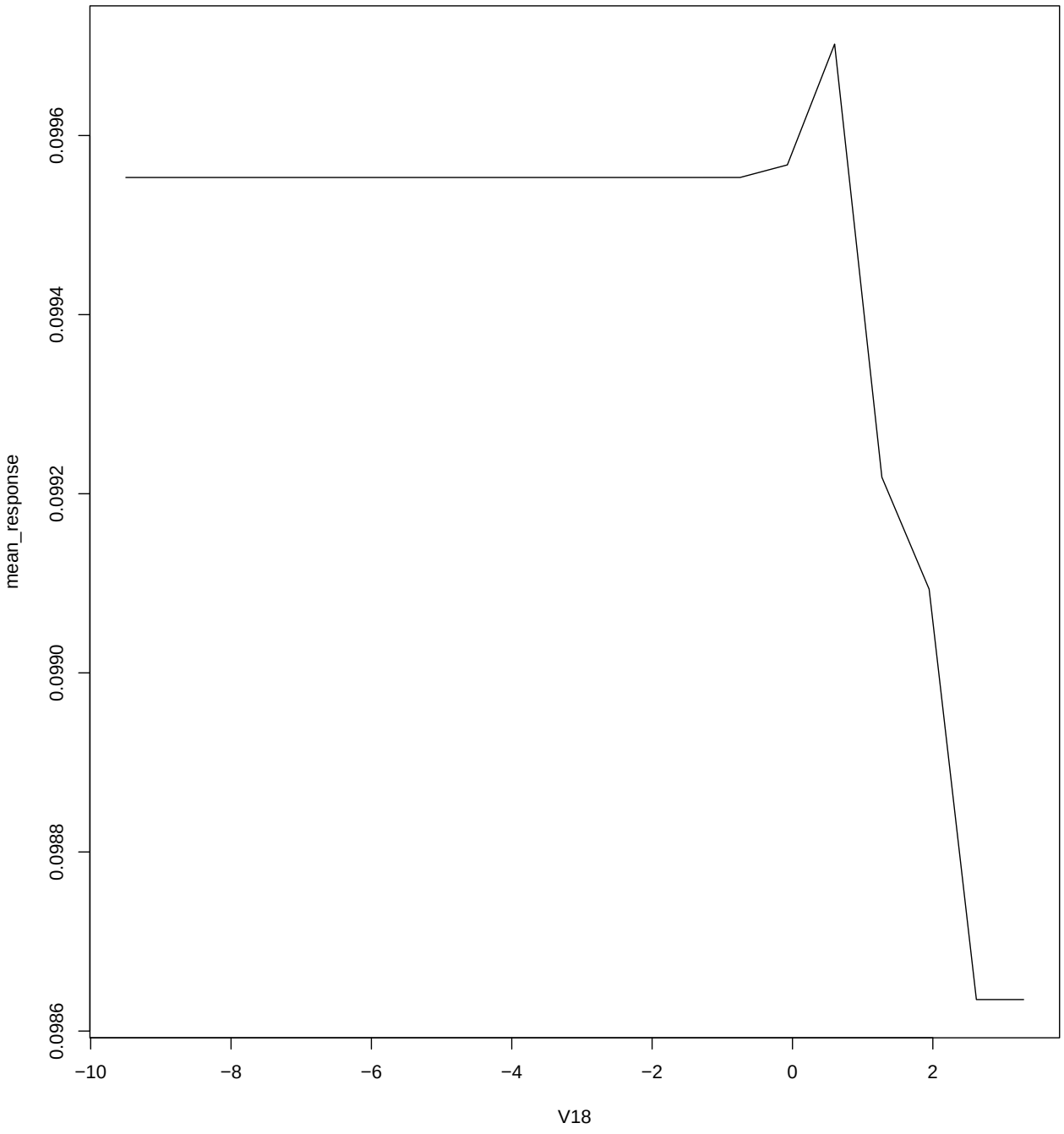

mean_response
0.102
0.101
0.100
0.099
0.098
0.097
−4
−2
0
2
4
V13

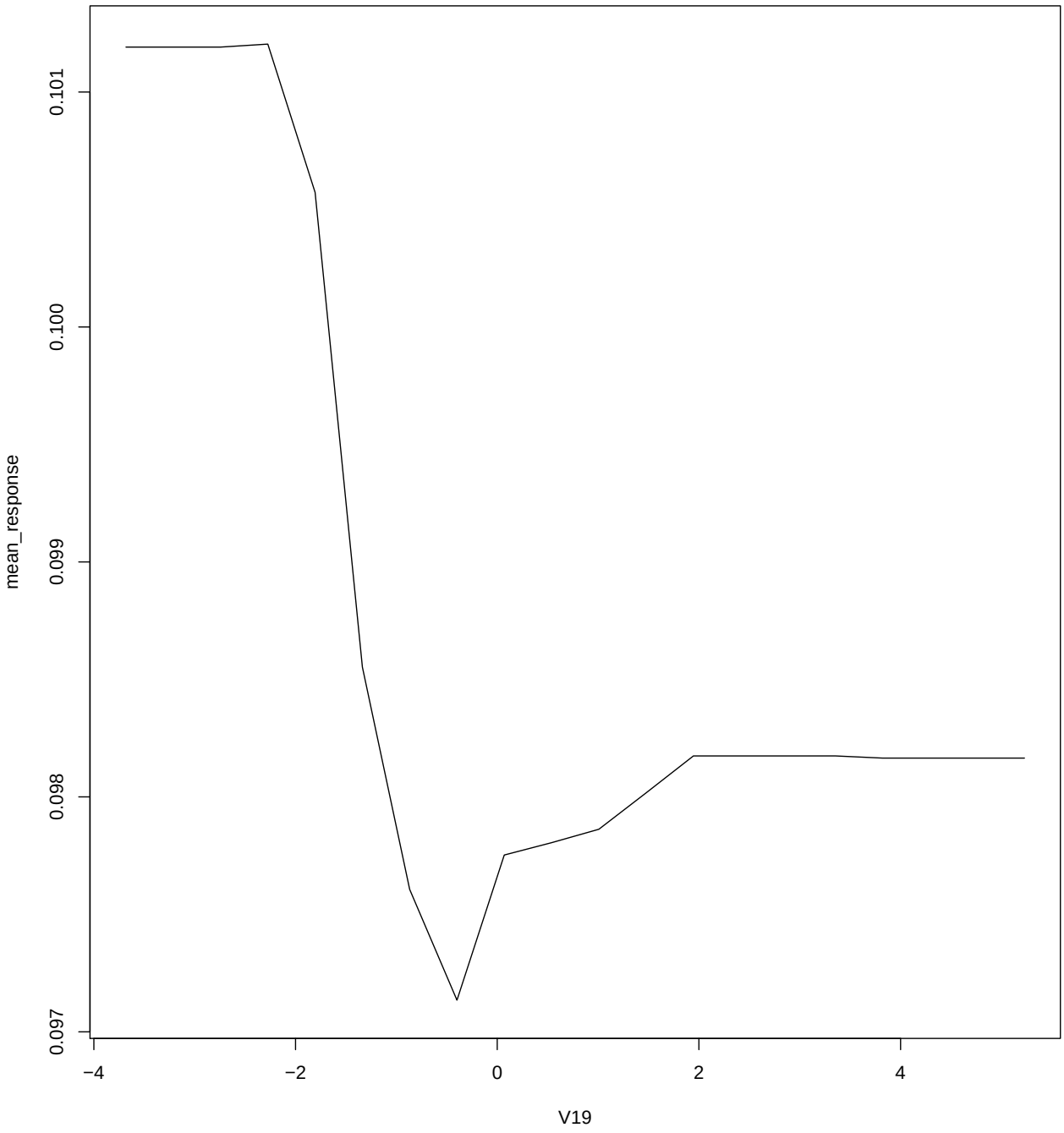

mean_response
0.1
0.2
0.3
0.4
−20
−15
−10
−5
0
5
V14

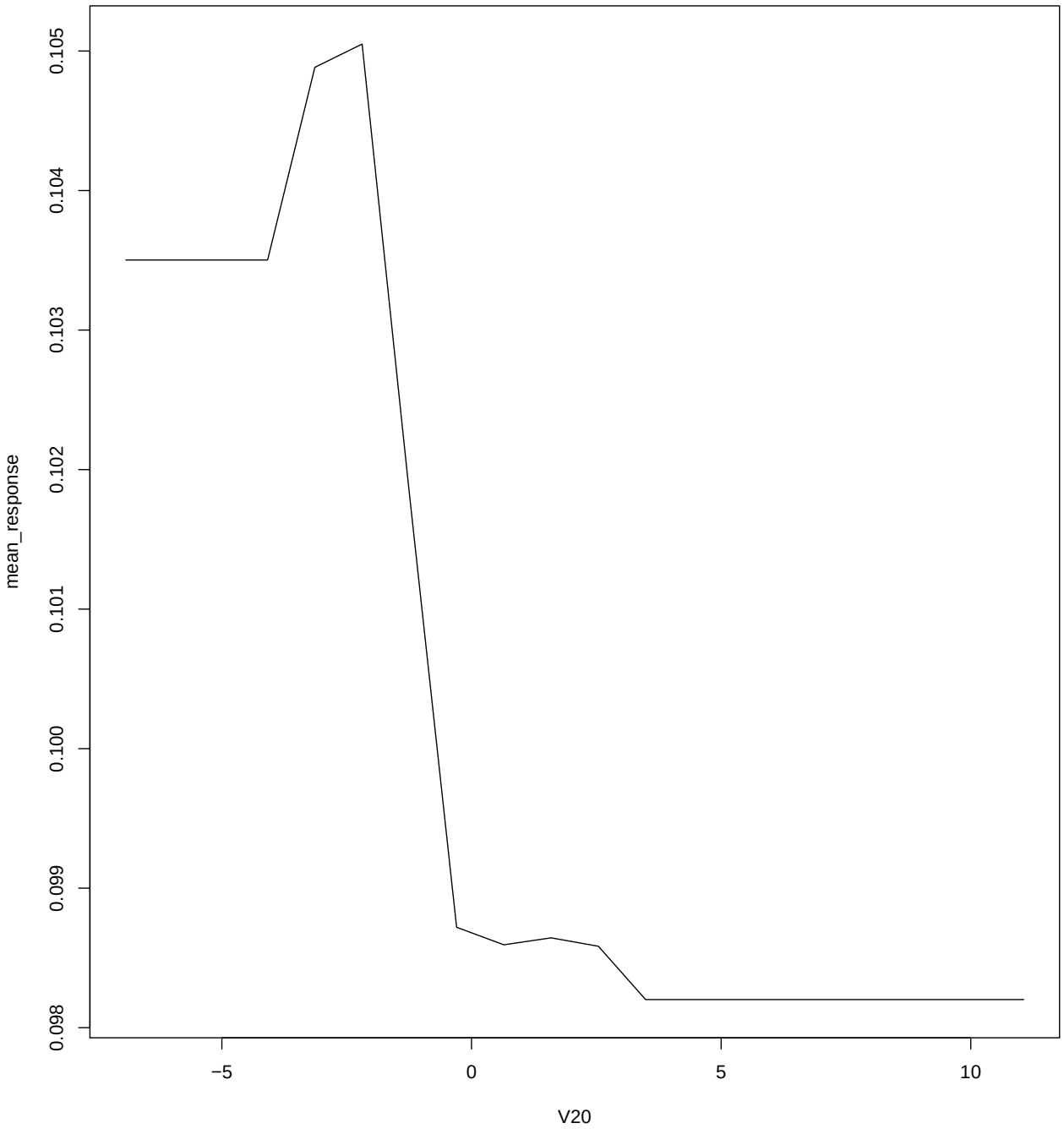

mean_response
0.098
0.100
0.102
0.104
0.106
0.108
0.110
−4
−2
0
2
V15

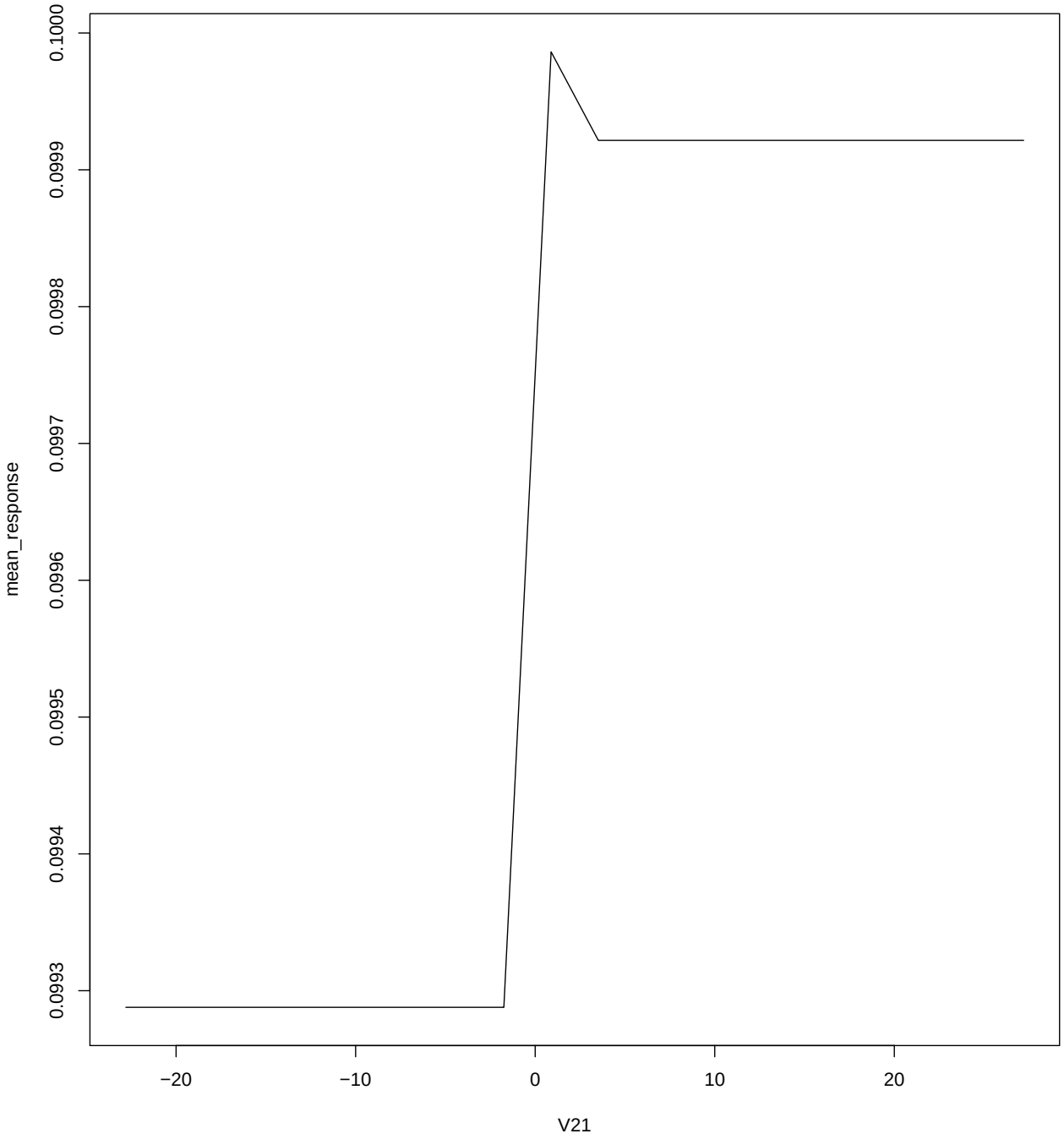

mean_response
0.0995
0.0990
0.0985
−10
−5
0
V16

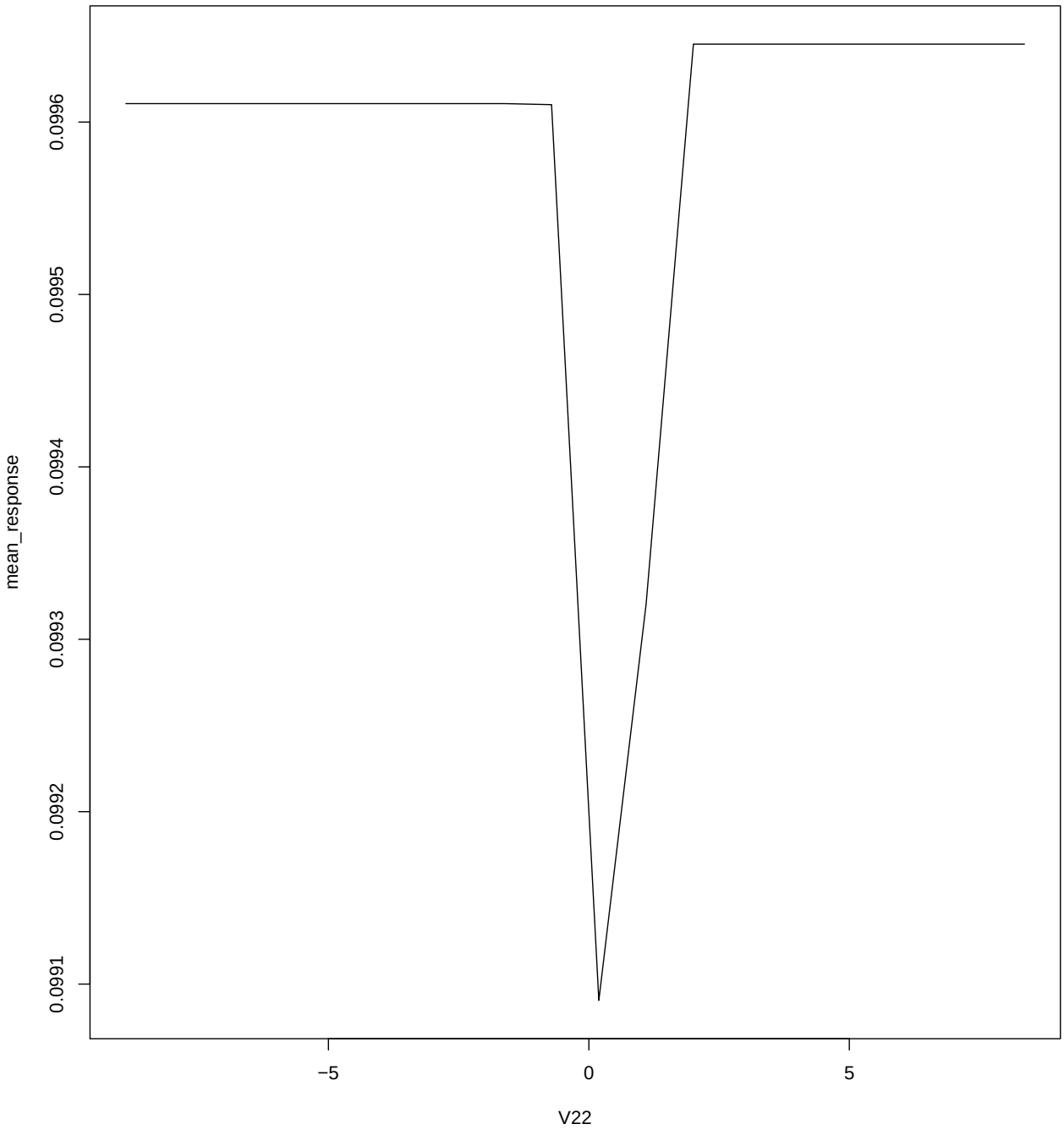

mean_response
0.100
0.105
0.110
0.115
−25
−20
−15
−10
−5
0
5
V17

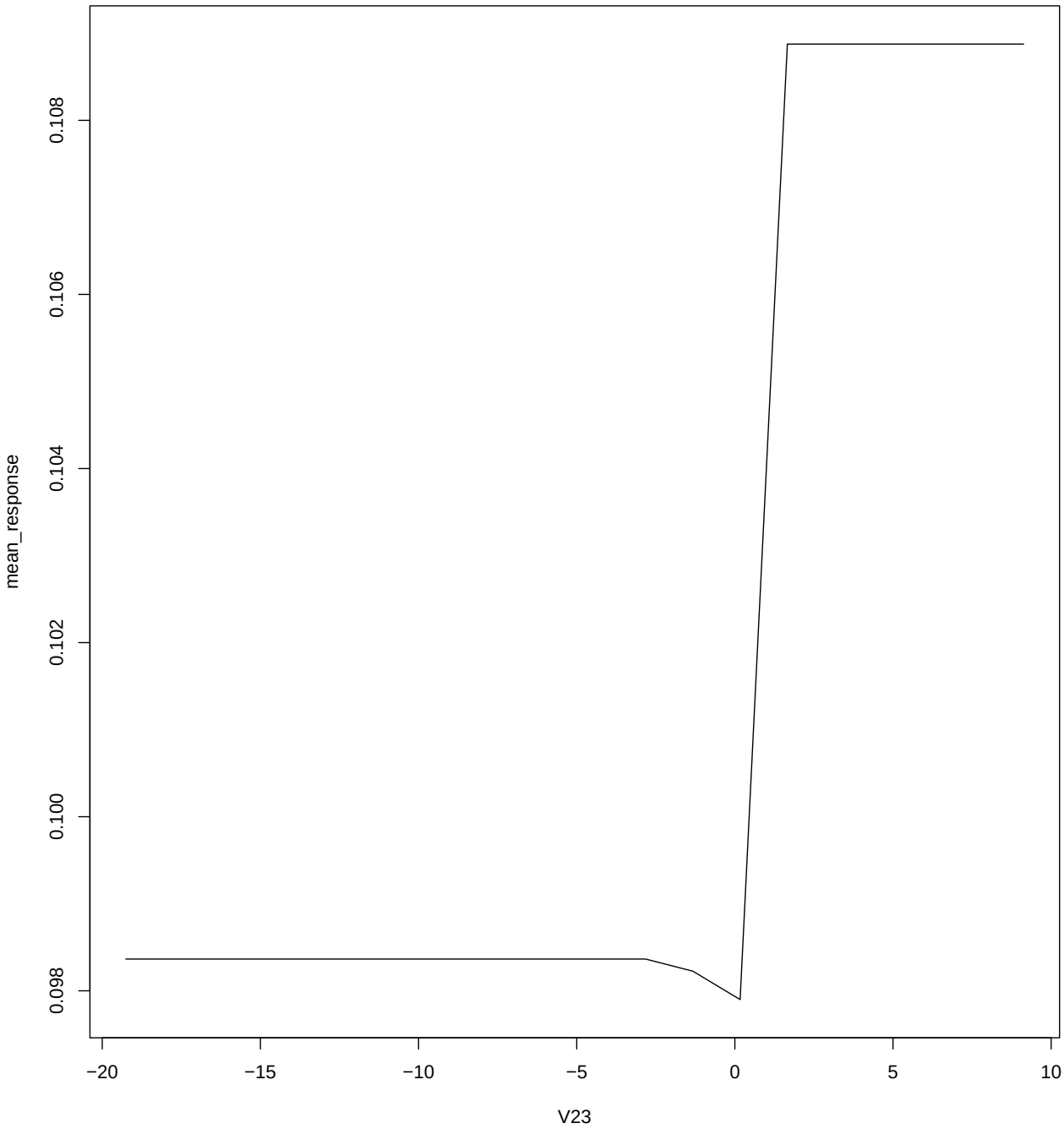

mean_response
0.0986
0.0988
0.0990
0.0992
0.0994
0.0996
−10
−8
−6
−4
−2
0
2
V18

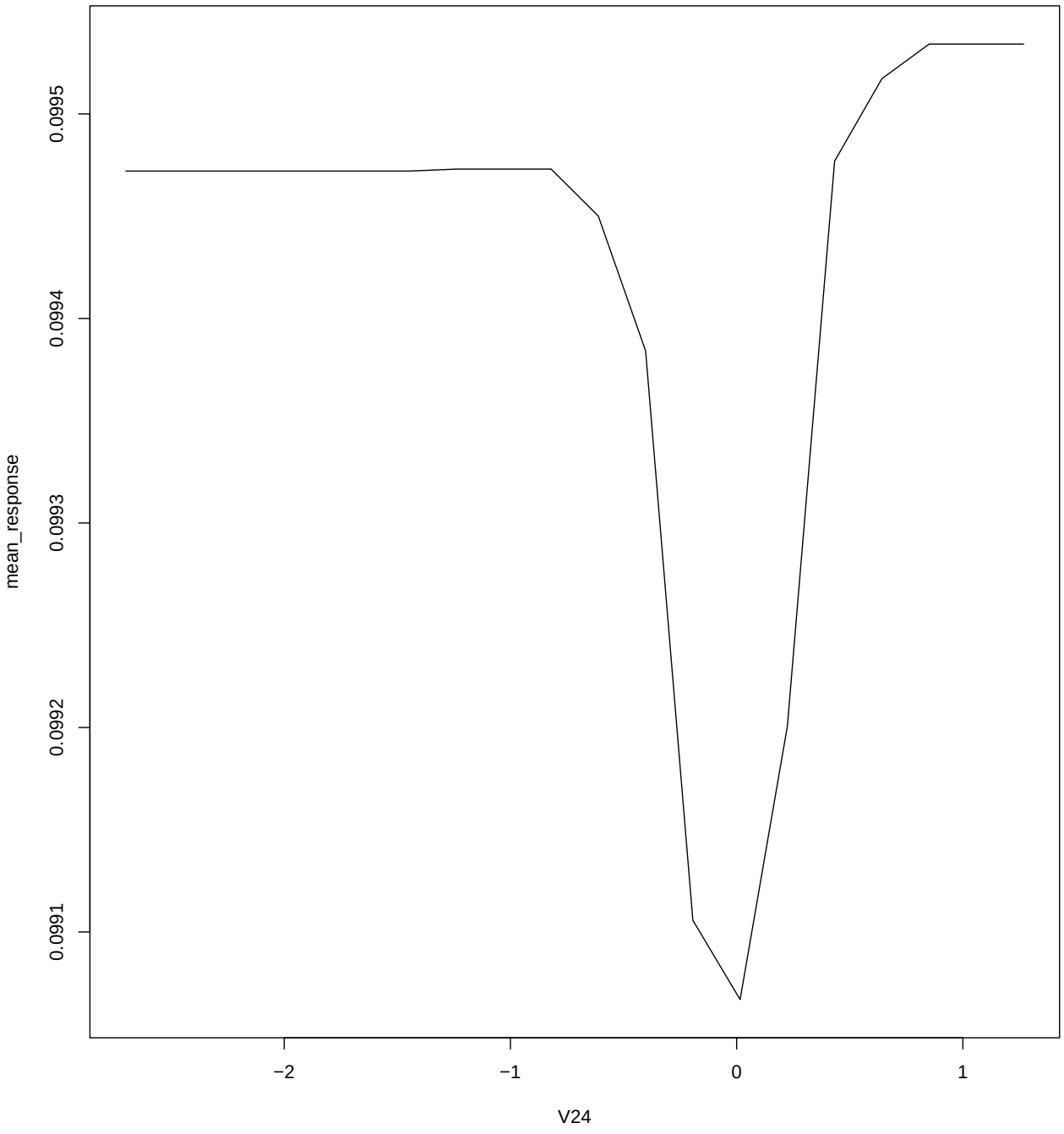

mean_response
0.097
0.098
0.099
0.100
0.101
−4
−2
0
2
4
V19

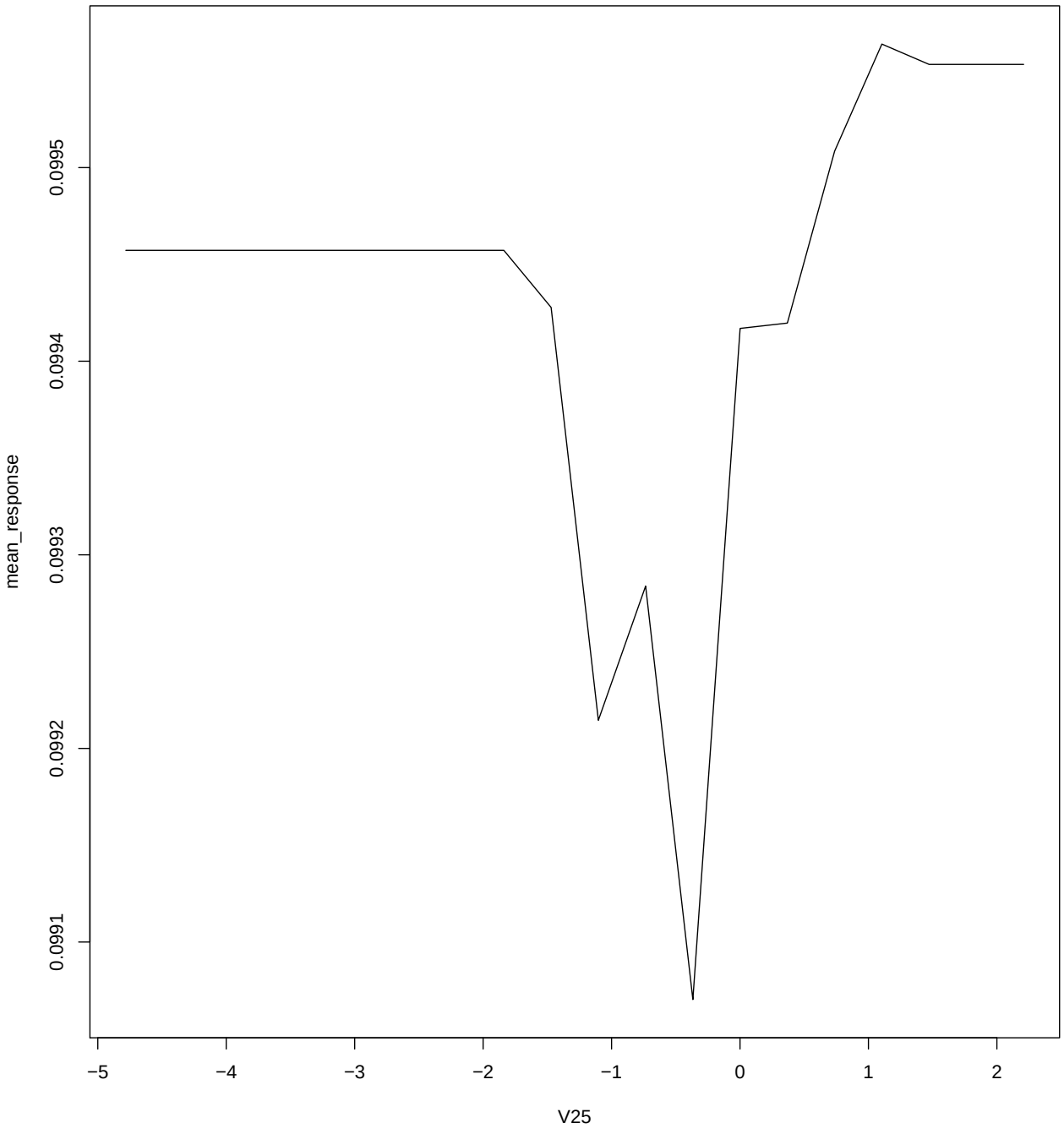

mean_response
0.098
0.099
0.100
0.101
0.102
0.103
0.104
0.105
−5
0
5
10
V20

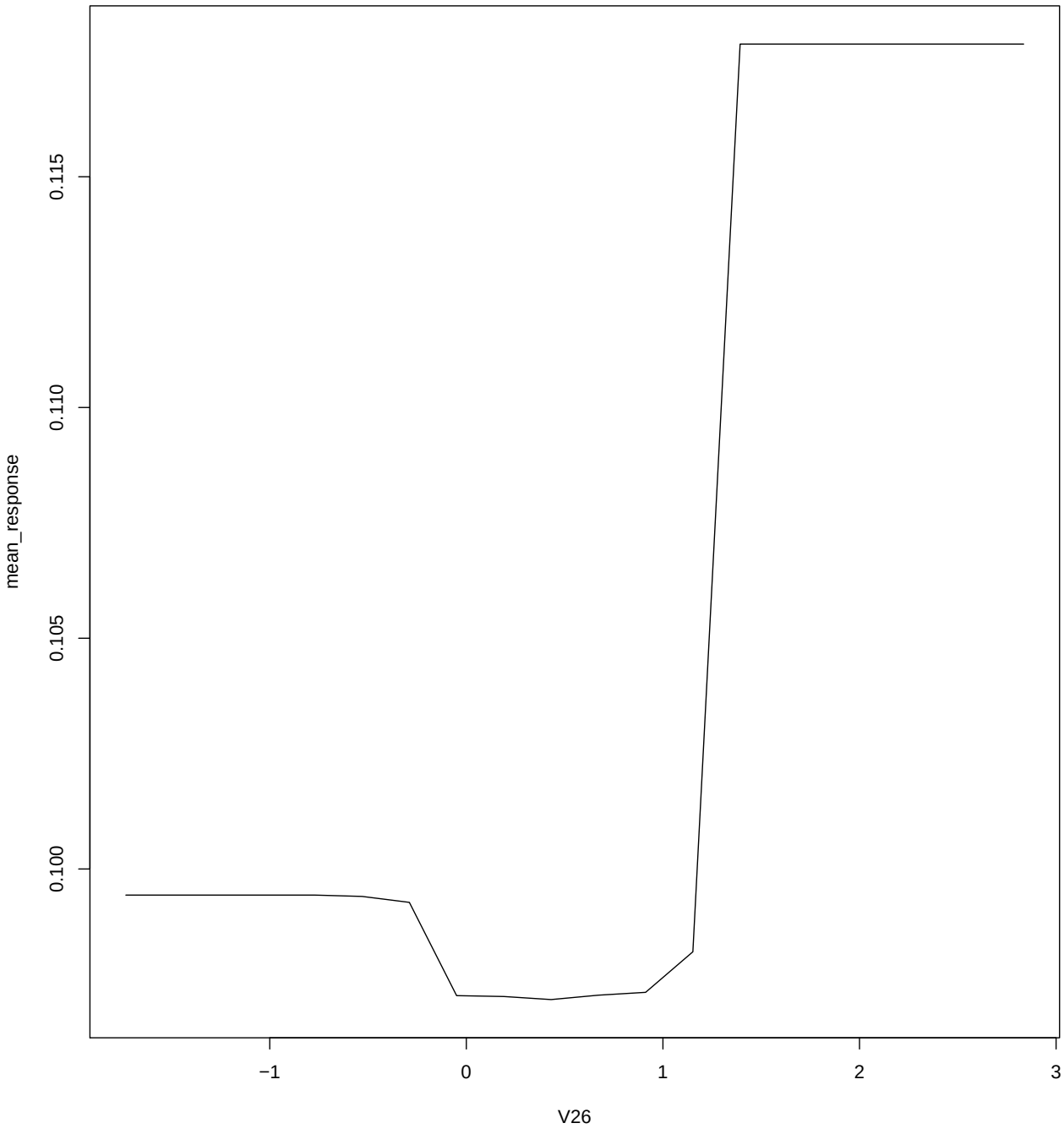

mean_response
0.0993
0.0994
0.0995
0.0996
0.0997
0.0998
0.0999
0.1000
−20
−10
0
10
20
V21

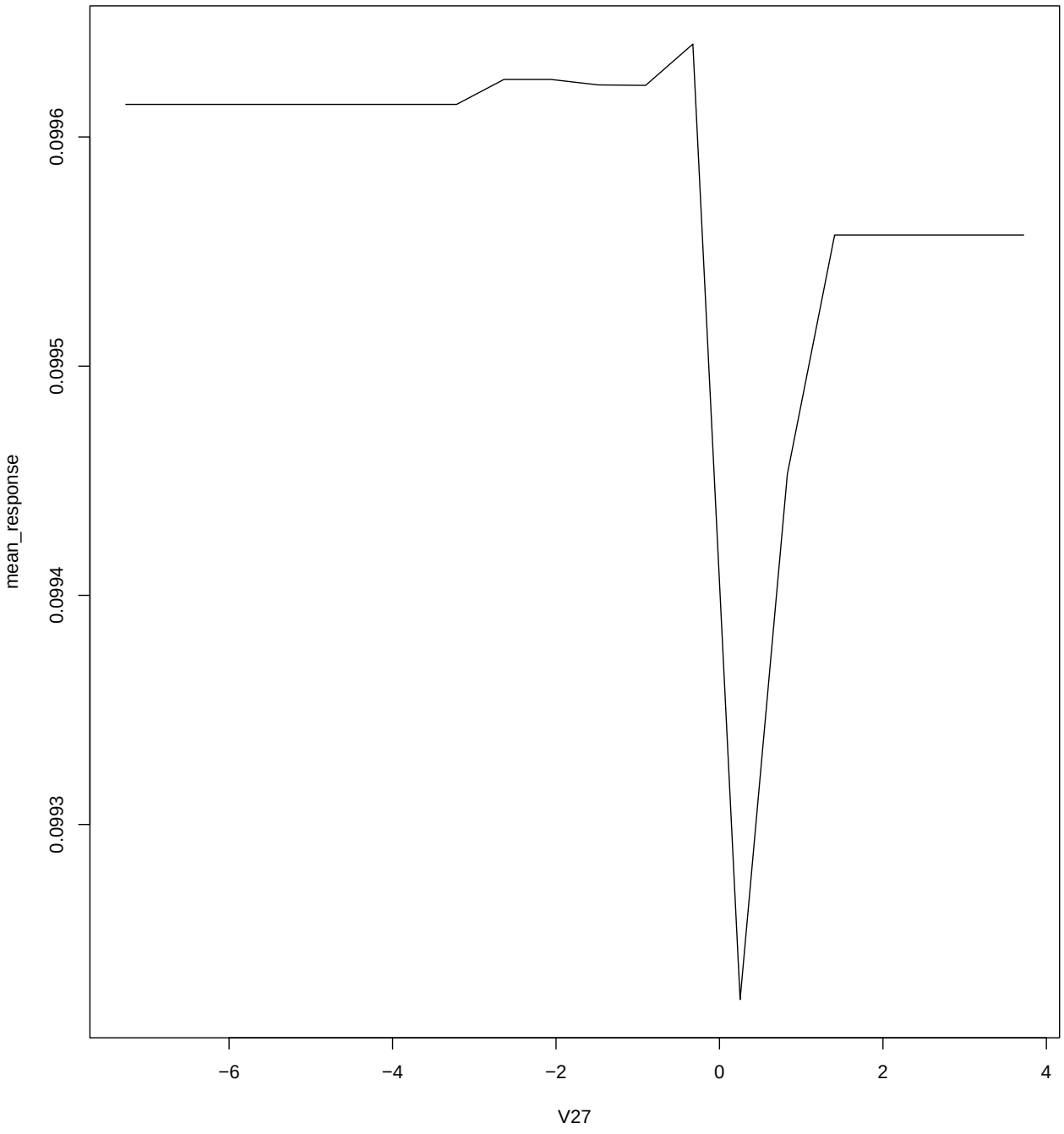

mean_response
0.0991
0.0992
0.0993
0.0994
0.0995
0.0996
−5
0
5
V22

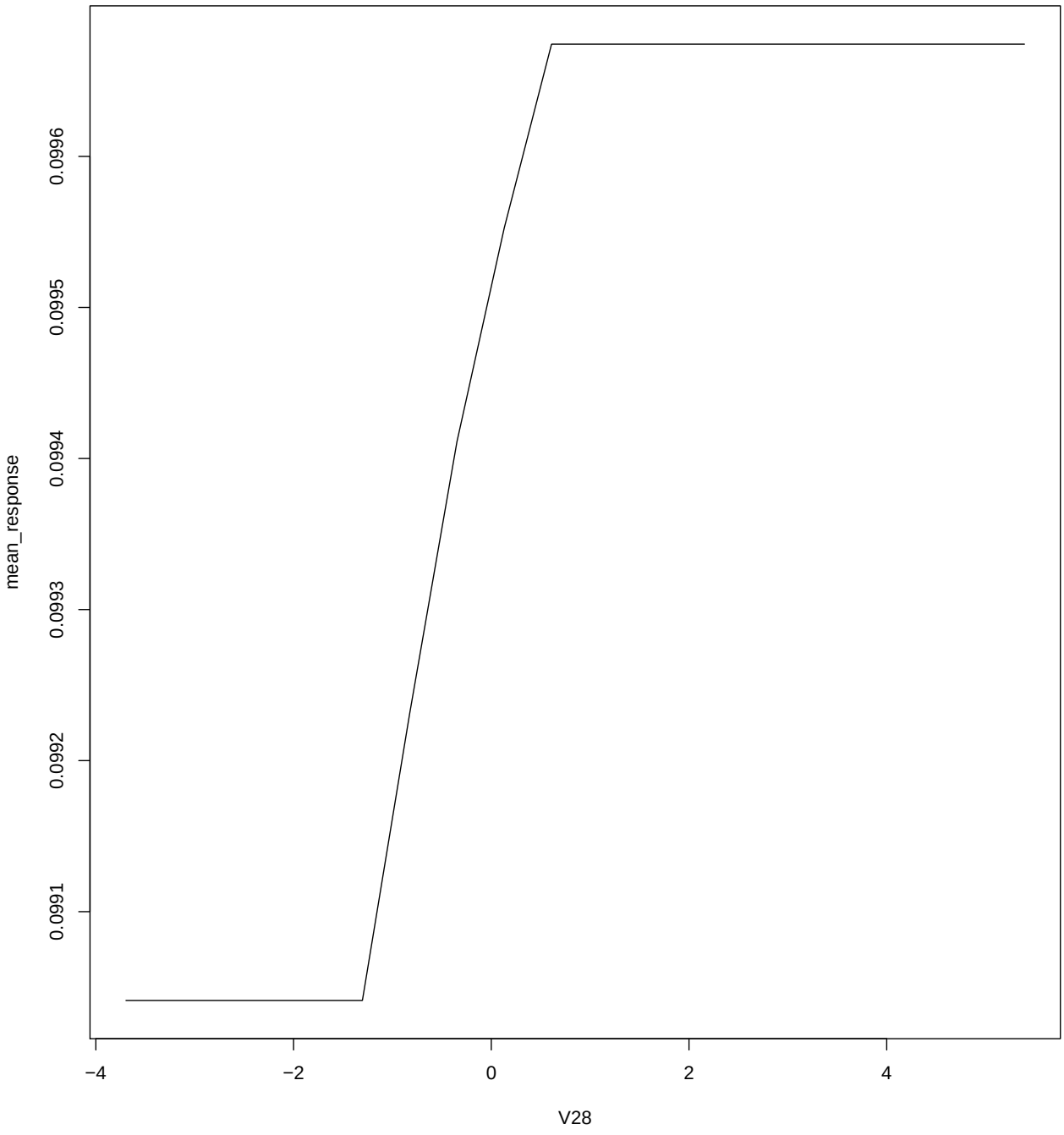

mean_response
0.098
0.100
0.102
0.104
0.106
0.108
−20
−15
−10
−5
0
5
10
V23

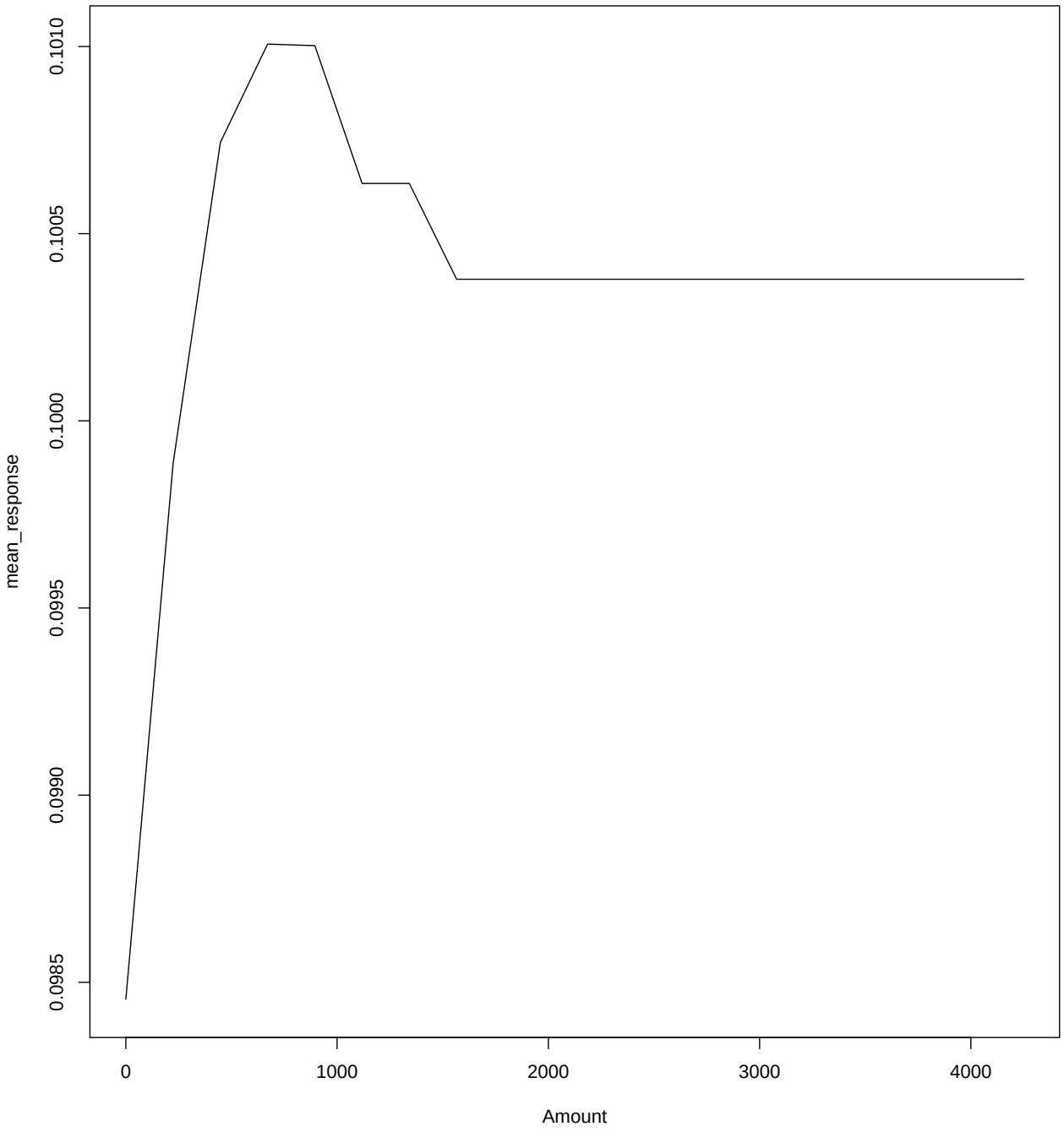

mean_response
0.0995
0.0994
0.0993
0.0992
0.0991
−2
−1
0
1
V24


mean_response
0.0995
0.0994
0.0993
0.0992
0.0991
−5
−4
−3
−2
−1
0
1
2
V25


mean_response
0.100
0.105
0.110
0.115
−1
0
1
2
3
V26


mean_response
0.0996
0.0995
0.0994
0.0993
−6
−4
−2
0
2
4
V27


mean_response
0.0991
0.0992
0.0993
0.0994
0.0995
0.0996
−4
−2
0
2
4
V28




```
## [[1]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'Time'
##             Time mean_response stddev_response
## 1      37.000000      0.099765        0.293928
## 2    9126.263158      0.099765        0.293928
## 3   18215.526316      0.099765        0.293927
## 4   27304.789474      0.099698        0.293829
## 5   36394.052632      0.099698        0.293829
## 6   45483.315789      0.099849        0.294059
## 7   54572.578947      0.099837        0.294097
## 8   63661.842105      0.099756        0.294086
## 9   72751.105263      0.099248        0.293485
```

```
## 10  81840.368421      0.098185        0.292218
## 11  90929.631579      0.098036        0.292046
## 12 100018.894737      0.097863        0.291802
## 13 109108.157895      0.097828        0.291765
## 14 118197.421053      0.097817        0.291746
## 15 127286.684211      0.097816        0.291739
## 16 136375.947368      0.097550        0.291236
## 17 145465.210526      0.097653        0.291363
## 18 154554.473684      0.097783        0.291516
## 19 163643.736842      0.097987        0.291727
## 20 172733.000000      0.097988        0.291718
##
## [[2]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V1'
##            V1 mean_response stddev_response
## 1  -30.552380      0.097907        0.291621
## 2  -28.819086      0.097907        0.291621
## 3  -27.085791      0.097907        0.291621
## 4  -25.352497      0.097907        0.291621
## 5  -23.619203      0.097907        0.291621
## 6  -21.885908      0.097907        0.291621
## 7  -20.152614      0.097907        0.291621
## 8  -18.419320      0.097907        0.291621
## 9  -16.686025      0.097907        0.291621
## 10 -14.952731      0.097907        0.291621
## 11 -13.219436      0.097907        0.291621
## 12 -11.486142      0.097907        0.291621
## 13  -9.752848      0.097907        0.291621
## 14  -8.019553      0.097907        0.291621
## 15  -6.286259      0.098470        0.292540
## 16  -4.552965      0.098911        0.293272
## 17  -2.819670      0.099214        0.293731
## 18  -1.086376      0.098991        0.293365
## 19   0.646918      0.099457        0.293979
## 20   2.380213      0.099819        0.294399
##
## [[3]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V2'
##            V2 mean_response stddev_response
## 1  -20.660417      0.098711        0.292754
## 2  -18.412094      0.098711        0.292754
## 3  -16.163770      0.098711        0.292754
## 4  -13.915447      0.098711        0.292754
## 5  -11.667123      0.098711        0.292754
## 6   -9.418800      0.098711        0.292754
## 7   -7.170476      0.098711        0.292754
## 8   -4.922153      0.098876        0.293069
## 9   -2.673829      0.099528        0.294396
## 10  -0.425506      0.099519        0.294427
## 11   1.822818      0.099645        0.294527
## 12   4.071141      0.099646        0.294531
## 13   6.319465      0.099646        0.294531
## 14   8.567788      0.099646        0.294531
## 15  10.816112      0.099646        0.294531
```

```
## 16  13.064435      0.099646        0.294531
## 17  15.312759      0.099646        0.294531
## 18  17.561082      0.099646        0.294531
## 19  19.809405      0.099646        0.294531
## 20  22.057729      0.099646        0.294531
##
## [[4]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V3'
##            V3 mean_response stddev_response
## 1  -31.103685      0.099609        0.293449
## 2  -29.269552      0.099609        0.293449
## 3  -27.435419      0.099609        0.293449
## 4  -25.601285      0.099609        0.293449
## 5  -23.767152      0.099609        0.293449
## 6  -21.933019      0.099609        0.293449
## 7  -20.098886      0.099609        0.293449
## 8  -18.264753      0.099609        0.293449
## 9  -16.430620      0.099609        0.293449
## 10 -14.596487      0.099609        0.293449
## 11 -12.762354      0.099609        0.293449
## 12 -10.928221      0.099609        0.293449
## 13  -9.094087      0.099609        0.293449
## 14  -7.259954      0.099609        0.293449
## 15  -5.425821      0.099609        0.293449
## 16  -3.591688      0.100086        0.294139
## 17  -1.757555      0.098893        0.293157
## 18   0.076578      0.098946        0.293193
## 19   1.910711      0.099322        0.293563
## 20   3.744844      0.099339        0.293562
##
## [[5]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V4'
##           V4 mean_response stddev_response
## 1  -4.964819      0.091799        0.281332
## 2  -4.076872      0.091799        0.281332
## 3  -3.188926      0.091799        0.281332
## 4  -2.300979      0.091799        0.281332
## 5  -1.413032      0.091901        0.281601
## 6  -0.525085      0.092671        0.282838
## 7   0.362862      0.095743        0.287335
## 8   1.250808      0.097026        0.288946
## 9   2.138755      0.098739        0.290197
## 10  3.026702      0.116307        0.292962
## 11  3.914649      0.111769        0.292497
## 12  4.802596      0.119667        0.294158
## 13  5.690542      0.128930        0.294932
## 14  6.578489      0.128930        0.294932
## 15  7.466436      0.129424        0.294888
## 16  8.354383      0.129424        0.294888
## 17  9.242330      0.129424        0.294888
## 18 10.130276      0.129424        0.294888
## 19 11.018223      0.129424        0.294888
## 20 11.906170      0.129424        0.294888
##
```

```
## [[6]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V5'
##            V5 mean_response stddev_response
## 1  -22.105532      0.098060        0.291603
## 2  -20.186525      0.098060        0.291603
## 3  -18.267518      0.098060        0.291603
## 4  -16.348511      0.098060        0.291603
## 5  -14.429504      0.098060        0.291603
## 6  -12.510497      0.098061        0.291604
## 7  -10.591491      0.098061        0.291604
## 8   -8.672484      0.098061        0.291604
## 9   -6.753477      0.098061        0.291604
## 10  -4.834470      0.098061        0.291604
## 11  -2.915463      0.098646        0.292568
## 12  -0.996457      0.098748        0.292743
## 13   0.922550      0.099549        0.294025
## 14   2.841557      0.099497        0.293957
## 15   4.760564      0.100137        0.294268
## 16   6.679571      0.101647        0.294726
## 17   8.598578      0.101647        0.294726
## 18  10.517584      0.101647        0.294726
## 19  12.436591      0.101647        0.294726
## 20  14.355598      0.101647        0.294726
##
## [[7]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V6'
##            V6 mean_response stddev_response
## 1  -11.656632      0.100403        0.294740
## 2  -10.357507      0.100403        0.294740
## 3   -9.058382      0.100403        0.294740
## 4   -7.759257      0.100403        0.294740
## 5   -6.460131      0.100403        0.294740
## 6   -5.161006      0.100403        0.294740
## 7   -3.861881      0.100240        0.294591
## 8   -2.562756      0.100109        0.294543
## 9   -1.263631      0.099935        0.294319
## 10   0.035495      0.099670        0.294000
## 11   1.334620      0.099183        0.292843
## 12   2.633745      0.097019        0.289597
## 13   3.932870      0.097019        0.289597
## 14   5.231995      0.097019        0.289597
## 15   6.531121      0.097019        0.289597
## 16   7.830246      0.097019        0.289597
## 17   9.129371      0.097019        0.289597
## 18  10.428496      0.097019        0.289597
## 19  11.727622      0.097019        0.289597
## 20  13.026747      0.097019        0.289597
##
## [[8]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V7'
##            V7 mean_response stddev_response
## 1  -43.557242      0.100494        0.292192
## 2  -40.252943      0.100494        0.292192
## 3  -36.948644      0.100494        0.292192
```

```
## 4  -33.644345      0.100494        0.292192
## 5  -30.340047      0.100494        0.292192
## 6  -27.035748      0.100494        0.292192
## 7  -23.731449      0.100494        0.292192
## 8  -20.427150      0.100494        0.292192
## 9  -17.122852      0.100494        0.292192
## 10 -13.818553      0.100494        0.292192
## 11 -10.514254      0.100494        0.292192
## 12  -7.209955      0.100494        0.292192
## 13  -3.905657      0.097206        0.290758
## 14  -0.601358      0.097199        0.290777
## 15   2.702941      0.103708        0.291472
## 16   6.007240      0.139996        0.284119
## 17   9.311538      0.139996        0.284119
## 18  12.615837      0.139996        0.284119
## 19  15.920136      0.139996        0.284119
## 20  19.224435      0.139996        0.284119
##
## [[9]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V8'
##            V8 mean_response stddev_response
## 1  -41.044261      0.100110        0.294920
## 2  -37.831026      0.100110        0.294920
## 3  -34.617790      0.100110        0.294920
## 4  -31.404555      0.100110        0.294920
## 5  -28.191320      0.100110        0.294920
## 6  -24.978085      0.100110        0.294920
## 7  -21.764850      0.100110        0.294920
## 8  -18.551614      0.100110        0.294920
## 9  -15.338379      0.100110        0.294920
## 10 -12.125144      0.100110        0.294920
## 11  -8.911909      0.100110        0.294920
## 12  -5.698673      0.100110        0.294920
## 13  -2.485438      0.100110        0.294919
## 14   0.727797      0.098421        0.292341
## 15   3.941032      0.098421        0.292341
## 16   7.154267      0.098421        0.292341
## 17  10.367503      0.098421        0.292341
## 18  13.580738      0.098421        0.292341
## 19  16.793973      0.098421        0.292341
## 20  20.007208      0.098421        0.292341
##
## [[10]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V9'
##            V9 mean_response stddev_response
## 1  -13.434066      0.099476        0.294154
## 2  -12.309221      0.099476        0.294154
## 3  -11.184376      0.099476        0.294154
## 4  -10.059531      0.099476        0.294154
## 5   -8.934686      0.099476        0.294154
## 6   -7.809841      0.099476        0.294154
## 7   -6.684996      0.099476        0.294154
## 8   -5.560151      0.099476        0.294154
## 9   -4.435306      0.099476        0.294154
```

```
## 10  -3.310461      0.099476        0.294154
## 11  -2.185616      0.099476        0.294154
## 12  -1.060771      0.099456        0.294140
## 13   0.064074      0.099429        0.294183
## 14   1.188919      0.099451        0.294134
## 15   2.313764      0.099446        0.294121
## 16   3.438609      0.099218        0.293665
## 17   4.563454      0.099218        0.293665
## 18   5.688299      0.099218        0.293665
## 19   6.813143      0.099218        0.293665
## 20   7.937988      0.099218        0.293665
##
## [[11]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V10'
##           V10 mean_response stddev_response
## 1  -24.588262      0.105943        0.294412
## 2  -22.687982      0.105943        0.294412
## 3  -20.787702      0.105943        0.294412
## 4  -18.887421      0.105943        0.294412
## 5  -16.987141      0.105943        0.294412
## 6  -15.086861      0.105943        0.294412
## 7  -13.186580      0.105943        0.294412
## 8  -11.286300      0.105943        0.294412
## 9   -9.386019      0.105943        0.294412
## 10  -7.485739      0.105943        0.294412
## 11  -5.585459      0.105943        0.294412
## 12  -3.685178      0.105944        0.294413
## 13  -1.784898      0.100171        0.292102
## 14   0.115383      0.096452        0.286609
## 15   2.015663      0.096225        0.285568
## 16   3.915943      0.095642        0.282699
## 17   5.816224      0.095642        0.282699
## 18   7.716504      0.095642        0.282699
## 19   9.616784      0.095642        0.282699
## 20  11.517065      0.095642        0.282699
##
## [[12]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V11'
##          V11 mean_response stddev_response
## 1  -3.318559      0.096012        0.285486
## 2  -2.511324      0.096012        0.285486
## 3  -1.704088      0.096012        0.285486
## 4  -0.896853      0.095906        0.285372
## 5  -0.089618      0.096623        0.286899
## 6   0.717618      0.098915        0.292641
## 7   1.524853      0.100906        0.294902
## 8   2.332089      0.100719        0.294700
## 9   3.139324      0.100719        0.294700
## 10  3.946559      0.100719        0.294699
## 11  4.753795      0.100719        0.294699
## 12  5.561030      0.100718        0.294699
## 13  6.368265      0.100718        0.294699
## 14  7.175501      0.100718        0.294699
## 15  7.982736      0.100718        0.294699
```

```
## 16  8.789972      0.100718        0.294699
## 17  9.597207      0.100718        0.294699
## 18 10.404442      0.100718        0.294699
## 19 11.211678      0.100718        0.294699
## 20 12.018913      0.100718        0.294699
##
## [[13]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V12'
##           V12 mean_response stddev_response
## 1  -18.683715      0.106310        0.294224
## 2  -17.485681      0.106310        0.294224
## 3  -16.287648      0.106310        0.294224
## 4  -15.089615      0.106310        0.294224
## 5  -13.891582      0.106310        0.294224
## 6  -12.693549      0.106310        0.294224
## 7  -11.495516      0.106310        0.294224
## 8  -10.297483      0.106310        0.294224
## 9   -9.099449      0.106310        0.294224
## 10  -7.901416      0.106310        0.294224
## 11  -6.703383      0.106310        0.294224
## 12  -5.505350      0.106310        0.294224
## 13  -4.307317      0.106096        0.294164
## 14  -3.109284      0.100877        0.293927
## 15  -1.911251      0.099413        0.292860
## 16  -0.713217      0.098713        0.291313
## 17   0.484816      0.094608        0.282790
## 18   1.682849      0.094603        0.282784
## 19   2.880882      0.094603        0.282784
## 20   4.078915      0.094603        0.282784
##
## [[14]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V13'
##          V13 mean_response stddev_response
## 1  -3.766274      0.102097        0.293794
## 2  -3.368359      0.102097        0.293794
## 3  -2.970444      0.102097        0.293794
## 4  -2.572529      0.102097        0.293794
## 5  -2.174614      0.102097        0.293794
## 6  -1.776699      0.101234        0.293559
## 7  -1.378784      0.098694        0.292253
## 8  -0.980869      0.098480        0.292147
## 9  -0.582954      0.097950        0.291596
## 10 -0.185039      0.097387        0.290922
## 11  0.212876      0.096774        0.290133
## 12  0.610791      0.096792        0.290169
## 13  1.008706      0.097105        0.290502
## 14  1.406621      0.097523        0.290864
## 15  1.804536      0.097523        0.290864
## 16  2.202451      0.097523        0.290864
## 17  2.600366      0.097523        0.290864
## 18  2.998281      0.097523        0.290864
## 19  3.396195      0.097523        0.290864
## 20  3.794110      0.097523        0.290864
##
```

```
## [[15]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V14'
##           V14 mean_response stddev_response
## 1  -19.214325      0.466697        0.252562
## 2  -17.857873      0.466697        0.252562
## 3  -16.501420      0.466697        0.252562
## 4  -15.144968      0.466697        0.252562
## 5  -13.788515      0.466697        0.252562
## 6  -12.432063      0.466697        0.252562
## 7  -11.075610      0.466697        0.252562
## 8   -9.719158      0.466697        0.252562
## 9   -8.362705      0.466697        0.252562
## 10  -7.006253      0.466697        0.252562
## 11  -5.649800      0.466699        0.252565
## 12  -4.293348      0.389789        0.260205
## 13  -2.936895      0.120403        0.286600
## 14  -1.580443      0.101618        0.284297
## 15  -0.223990      0.081955        0.258351
## 16   1.132462      0.079241        0.251669
## 17   2.488915      0.079660        0.252241
## 18   3.845367      0.079660        0.252241
## 19   5.201820      0.079660        0.252241
## 20   6.558272      0.079660        0.252241
##
## [[16]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V15'
##          V15 mean_response stddev_response
## 1  -4.498945      0.098949        0.293314
## 2  -4.107963      0.098949        0.293314
## 3  -3.716982      0.098949        0.293314
## 4  -3.326000      0.098949        0.293314
## 5  -2.935019      0.098949        0.293314
## 6  -2.544037      0.098949        0.293314
## 7  -2.153056      0.098949        0.293314
## 8  -1.762074      0.098953        0.293319
## 9  -1.371093      0.098898        0.293234
## 10 -0.980111      0.098803        0.293107
## 11 -0.589130      0.098376        0.292450
## 12 -0.198148      0.098487        0.292679
## 13  0.192833      0.098329        0.292475
## 14  0.583815      0.098122        0.292112
## 15  0.974796      0.098379        0.292280
## 16  1.365778      0.098988        0.292900
## 17  1.756760      0.099162        0.293053
## 18  2.147741      0.106170        0.293955
## 19  2.538723      0.110140        0.292414
## 20  2.929704      0.110140        0.292414
##
## [[17]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V16'
##           V16 mean_response stddev_response
## 1  -14.129855      0.098900        0.293252
## 2  -13.218648      0.098900        0.293252
## 3  -12.307442      0.098900        0.293252
```

```
## 4  -11.396236      0.098900        0.293251
## 5  -10.485030      0.098900        0.293251
## 6   -9.573824      0.098900        0.293251
## 7   -8.662618      0.098900        0.293251
## 8   -7.751412      0.098900        0.293251
## 9   -6.840206      0.098900        0.293251
## 10  -5.929000      0.098900        0.293251
## 11  -5.017794      0.098900        0.293251
## 12  -4.106588      0.098900        0.293251
## 13  -3.195382      0.098900        0.293251
## 14  -2.284176      0.098893        0.293245
## 15  -1.372970      0.098822        0.293142
## 16  -0.461764      0.098563        0.292744
## 17   0.449442      0.098307        0.292252
## 18   1.360648      0.099765        0.293079
## 19   2.271854      0.099498        0.291754
## 20   3.183060      0.099498        0.291754
##
## [[18]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V17'
##           V17 mean_response stddev_response
## 1  -25.162799      0.116235        0.294290
## 2  -23.483737      0.116235        0.294290
## 3  -21.804675      0.116235        0.294290
## 4  -20.125612      0.116235        0.294290
## 5  -18.446550      0.116235        0.294290
## 6  -16.767488      0.116235        0.294290
## 7  -15.088426      0.116235        0.294290
## 8  -13.409363      0.116235        0.294290
## 9  -11.730301      0.116235        0.294290
## 10 -10.051239      0.116235        0.294290
## 11  -8.372176      0.116235        0.294290
## 12  -6.693114      0.116235        0.294290
## 13  -5.014052      0.116235        0.294290
## 14  -3.334989      0.116209        0.294282
## 15  -1.655927      0.100368        0.292959
## 16   0.023135      0.098574        0.291647
## 17   1.702197      0.099788        0.291965
## 18   3.381260      0.098881        0.289931
## 19   5.060322      0.099695        0.290396
## 20   6.739384      0.099695        0.290396
##
## [[19]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V18'
##          V18 mean_response stddev_response
## 1  -9.498746      0.099553        0.294167
## 2  -8.825460      0.099553        0.294167
## 3  -8.152174      0.099553        0.294167
## 4  -7.478888      0.099553        0.294167
## 5  -6.805602      0.099553        0.294167
## 6  -6.132316      0.099553        0.294167
## 7  -5.459030      0.099553        0.294167
## 8  -4.785744      0.099553        0.294167
## 9  -4.112458      0.099553        0.294167
```

```
## 10 -3.439172      0.099553        0.294167
## 11 -2.765886      0.099553        0.294167
## 12 -2.092600      0.099553        0.294167
## 13 -1.419314      0.099553        0.294167
## 14 -0.746028      0.099553        0.294167
## 15 -0.072742      0.099567        0.294242
## 16  0.600544      0.099702        0.294522
## 17  1.273830      0.099218        0.293684
## 18  1.947116      0.099093        0.293416
## 19  2.620402      0.098635        0.292687
## 20  3.293688      0.098635        0.292687
##
## [[20]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V19'
##          V19 mean_response stddev_response
## 1  -3.681904      0.101191        0.293236
## 2  -3.212943      0.101191        0.293236
## 3  -2.743983      0.101191        0.293236
## 4  -2.275023      0.101204        0.293240
## 5  -1.806062      0.100573        0.292942
## 6  -1.337102      0.098552        0.291777
## 7  -0.868142      0.097606        0.290735
## 8  -0.399182      0.097135        0.290133
## 9   0.069779      0.097753        0.291047
## 10  0.538739      0.097805        0.291152
## 11  1.007699      0.097862        0.291239
## 12  1.476660      0.098017        0.291404
## 13  1.945620      0.098174        0.291471
## 14  2.414580      0.098174        0.291471
## 15  2.883540      0.098174        0.291471
## 16  3.352501      0.098174        0.291471
## 17  3.821461      0.098165        0.291449
## 18  4.290421      0.098165        0.291449
## 19  4.759382      0.098165        0.291449
## 20  5.228342      0.098165        0.291449
##
## [[21]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V20'
##          V20 mean_response stddev_response
## 1  -6.923161      0.103502        0.292262
## 2  -5.976731      0.103502        0.292262
## 3  -5.030302      0.103502        0.292262
## 4  -4.083872      0.103502        0.292262
## 5  -3.137442      0.104883        0.293423
## 6  -2.191012      0.105050        0.293458
## 7  -1.244582      0.101829        0.294730
## 8  -0.298153      0.098720        0.293110
## 9   0.648277      0.098594        0.292810
## 10  1.594707      0.098644        0.292822
## 11  2.541137      0.098584        0.292736
## 12  3.487566      0.098202        0.292153
## 13  4.433996      0.098202        0.292153
## 14  5.380426      0.098202        0.292153
## 15  6.326856      0.098202        0.292153
```

```
## 16  7.273285      0.098202        0.292153
## 17  8.219715      0.098202        0.292153
## 18  9.166145      0.098202        0.292153
## 19 10.112575      0.098202        0.292153
## 20 11.059004      0.098202        0.292153
##
## [[22]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V21'
##           V21 mean_response stddev_response
## 1  -22.797604      0.099288        0.293855
## 2  -20.166002      0.099288        0.293855
## 3  -17.534399      0.099288        0.293855
## 4  -14.902797      0.099288        0.293855
## 5  -12.271195      0.099288        0.293855
## 6   -9.639593      0.099288        0.293855
## 7   -7.007990      0.099288        0.293855
## 8   -4.376388      0.099288        0.293855
## 9   -1.744786      0.099288        0.293855
## 10   0.886816      0.099986        0.294448
## 11   3.518419      0.099922        0.294371
## 12   6.150021      0.099922        0.294371
## 13   8.781623      0.099922        0.294371
## 14  11.413226      0.099922        0.294371
## 15  14.044828      0.099922        0.294371
## 16  16.676430      0.099922        0.294371
## 17  19.308032      0.099922        0.294371
## 18  21.939635      0.099922        0.294371
## 19  24.571237      0.099922        0.294371
## 20  27.202839      0.099922        0.294371
##
## [[23]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V22'
##          V22 mean_response stddev_response
## 1  -8.887017      0.099611        0.294442
## 2  -7.979175      0.099611        0.294442
## 3  -7.071333      0.099611        0.294442
## 4  -6.163490      0.099611        0.294442
## 5  -5.255648      0.099611        0.294442
## 6  -4.347806      0.099611        0.294442
## 7  -3.439964      0.099611        0.294442
## 8  -2.532122      0.099611        0.294442
## 9  -1.624279      0.099611        0.294442
## 10 -0.716437      0.099610        0.294468
## 11  0.191405      0.099091        0.293632
## 12  1.099247      0.099320        0.293932
## 13  2.007090      0.099645        0.294216
## 14  2.914932      0.099645        0.294216
## 15  3.822774      0.099645        0.294216
## 16  4.730616      0.099645        0.294216
## 17  5.638459      0.099645        0.294216
## 18  6.546301      0.099645        0.294216
## 19  7.454143      0.099645        0.294216
## 20  8.361985      0.099645        0.294216
##
```

```
## [[24]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V23'
##           V23 mean_response stddev_response
## 1  -19.254328      0.098365        0.291619
## 2  -17.760437      0.098365        0.291619
## 3  -16.266546      0.098365        0.291619
## 4  -14.772656      0.098365        0.291619
## 5  -13.278765      0.098365        0.291619
## 6  -11.784875      0.098365        0.291619
## 7  -10.290984      0.098365        0.291619
## 8   -8.797093      0.098365        0.291619
## 9   -7.303203      0.098365        0.291619
## 10  -5.809312      0.098365        0.291619
## 11  -4.315422      0.098365        0.291619
## 12  -2.821531      0.098365        0.291619
## 13  -1.327641      0.098225        0.291692
## 14   0.166250      0.097899        0.291508
## 15   1.660141      0.108876        0.293023
## 16   3.154031      0.108876        0.293023
## 17   4.647922      0.108876        0.293023
## 18   6.141812      0.108876        0.293023
## 19   7.635703      0.108876        0.293023
## 20   9.129594      0.108876        0.293023
##
## [[25]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V24'
##          V24 mean_response stddev_response
## 1  -2.700095      0.099472        0.294234
## 2  -2.491212      0.099472        0.294234
## 3  -2.282330      0.099472        0.294234
## 4  -2.073447      0.099472        0.294234
## 5  -1.864565      0.099472        0.294234
## 6  -1.655682      0.099472        0.294234
## 7  -1.446799      0.099472        0.294234
## 8  -1.237917      0.099473        0.294235
## 9  -1.029034      0.099473        0.294235
## 10 -0.820152      0.099473        0.294235
## 11 -0.611269      0.099450        0.294228
## 12 -0.402386      0.099384        0.294095
## 13 -0.193504      0.099106        0.293659
## 14  0.015379      0.099067        0.293593
## 15  0.224261      0.099201        0.293807
## 16  0.433144      0.099477        0.294193
## 17  0.642027      0.099517        0.294252
## 18  0.850909      0.099534        0.294271
## 19  1.059792      0.099534        0.294271
## 20  1.268675      0.099534        0.294271
##
## [[26]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V25'
##          V25 mean_response stddev_response
## 1  -4.781606      0.099457        0.293828
## 2  -4.413721      0.099457        0.293828
## 3  -4.045836      0.099457        0.293828
```

```
## 4  -3.677951      0.099457        0.293828
## 5  -3.310066      0.099457        0.293828
## 6  -2.942181      0.099457        0.293828
## 7  -2.574296      0.099457        0.293828
## 8  -2.206411      0.099457        0.293828
## 9  -1.838526      0.099457        0.293828
## 10 -1.470641      0.099428        0.293822
## 11 -1.102756      0.099215        0.293756
## 12 -0.734871      0.099284        0.293842
## 13 -0.366986      0.099070        0.293581
## 14  0.000899      0.099417        0.294242
## 15  0.368784      0.099420        0.294233
## 16  0.736669      0.099509        0.294343
## 17  1.104554      0.099564        0.294377
## 18  1.472439      0.099553        0.294358
## 19  1.840324      0.099553        0.294358
## 20  2.208209      0.099553        0.294358
##
## [[27]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V26'
##          V26 mean_response stddev_response
## 1  -1.732008      0.099434        0.294089
## 2  -1.491645      0.099434        0.294089
## 3  -1.251282      0.099434        0.294089
## 4  -1.010919      0.099434        0.294089
## 5  -0.770557      0.099434        0.294089
## 6  -0.530194      0.099405        0.294072
## 7  -0.289831      0.099276        0.293953
## 8  -0.049468      0.097255        0.290778
## 9   0.190895      0.097236        0.290726
## 10  0.431258      0.097170        0.290629
## 11  0.671621      0.097265        0.290716
## 12  0.911984      0.097327        0.290779
## 13  1.152347      0.098205        0.291604
## 14  1.392710      0.117875        0.291436
## 15  1.633073      0.117875        0.291436
## 16  1.873436      0.117875        0.291436
## 17  2.113799      0.117875        0.291436
## 18  2.354162      0.117875        0.291436
## 19  2.594524      0.117875        0.291436
## 20  2.834887      0.117875        0.291436
##
## [[28]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V27'
##          V27 mean_response stddev_response
## 1  -7.263482      0.099614        0.294582
## 2  -6.685329      0.099614        0.294582
## 3  -6.107175      0.099614        0.294582
## 4  -5.529021      0.099614        0.294582
## 5  -4.950868      0.099614        0.294582
## 6  -4.372714      0.099614        0.294582
## 7  -3.794560      0.099614        0.294582
## 8  -3.216407      0.099614        0.294582
## 9  -2.638253      0.099625        0.294607
```

```
## 10 -2.060099      0.099625        0.294607
## 11 -1.481946      0.099623        0.294602
## 12 -0.903792      0.099623        0.294601
## 13 -0.325638      0.099641        0.294652
## 14  0.252515      0.099224        0.293876
## 15  0.830669      0.099453        0.294133
## 16  1.408822      0.099557        0.294229
## 17  1.986976      0.099557        0.294229
## 18  2.565130      0.099557        0.294229
## 19  3.143283      0.099557        0.294229
## 20  3.721437      0.099557        0.294229
##
## [[29]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'V28'
##          V28 mean_response stddev_response
## 1  -3.695480      0.099041        0.293569
## 2  -3.217125      0.099041        0.293569
## 3  -2.738770      0.099041        0.293569
## 4  -2.260415      0.099041        0.293569
## 5  -1.782060      0.099041        0.293569
## 6  -1.303705      0.099041        0.293569
## 7  -0.825350      0.099231        0.293913
## 8  -0.346995      0.099411        0.294163
## 9   0.131360      0.099553        0.294475
## 10  0.609715      0.099674        0.294554
## 11  1.088070      0.099674        0.294554
## 12  1.566425      0.099674        0.294554
## 13  2.044780      0.099674        0.294554
## 14  2.523135      0.099674        0.294554
## 15  3.001490      0.099674        0.294554
## 16  3.479845      0.099674        0.294554
## 17  3.958200      0.099674        0.294554
## 18  4.436555      0.099674        0.294554
## 19  4.914910      0.099674        0.294554
## 20  5.393265      0.099674        0.294554
##
## [[30]]
## PartialDependence: Partial Dependence Plot of model gbm_grid_model_42 on column 'Amount'
##         Amount mean_response stddev_response
## 1     0.000000      0.098455        0.292553
## 2   223.670000      0.099886        0.294131
## 3   447.340000      0.100744        0.294634
## 4   671.010000      0.101006        0.294705
## 5   894.680000      0.101002        0.294695
## 6  1118.350000      0.100634        0.294205
## 7  1342.020000      0.100634        0.294205
## 8  1565.690000      0.100378        0.293751
## 9  1789.360000      0.100378        0.293751
## 10 2013.030000      0.100378        0.293751
## 11 2236.700000      0.100378        0.293751
## 12 2460.370000      0.100378        0.293751
## 13 2684.040000      0.100378        0.293751
## 14 2907.710000      0.100378        0.293751
## 15 3131.380000      0.100378        0.293751
```

```
## 16 3355.050000      0.100378        0.293751
## 17 3578.720000      0.100378        0.293751
## 18 3802.390000      0.100378        0.293751
## 19 4026.060000      0.100378        0.293751
## 20 4249.730000      0.100378        0.293751
```

```
#Partial dependence plot gives a graphical depiction of
#the marginal effect of a variable on the response.
#The effect of a variable is measured in change in the mean response.

# All done. Shut down H2O.
#h2o.shutdown(prompt = FALSE)
```

Variable Importance: In the variable importance graph, the top variables contribute more to the model than the bottom ones. Here, V14 and V10 are the two most significant variables. Too many variables can result in overfitting. In order to mitigate overfitting I could reduce the number of variables used to only the most important variables.

Partial Dependence: Partial dependence plots gives a graphical depiction of the marginal effect of a variable on the response (fraud in our case). The effect of a variable is measured in change in the mean response.