Probability of Recession

William Chiu

2022-11-21

Summary

Forecast the probability of a recession in the next 6 months using the following predictors:

1. Spread between 10Y CMT and Effective Federal Funds Rate
2. YOY change in Unemployment Rate
3. YOY growth in CPI-U
4. YOY change in Effective Federal Funds Rate
5. Adstock transformations of predictors

Extract Historical Data

Refer to this vignette for FRED data access.

```
library(tidyverse)
library(lubridate)
library(scam)
library(fredr)
library(effects)
library(car)
library(MLmetrics)
library(caret)
library(pdp)
library(gridExtra)
library(mboost)
library(gbm)
library(import)
library(randomForest)
library(glmnet)
library(gtsummary)
```

```
series_id <- c("FEDFUNDS", "GS10", "USREC", "UNRATE", "CPIAUCSL")

full_data <- map_dfr(series_id, function(x) {
  fredr(
    series_id = x,
    observation_start = as.Date("1950-01-01"),
    observation_end = as.Date("2022-12-01")
  )
})
```

Pivot Wider

```
full_data_wide_raw <- full_data %>%
  arrange(date) %>%
  select(date, series_id, value) %>%
  pivot_wider(id_cols=date, names_from = series_id,
              values_from = value)
```

Calculate Features/Predictors

```
full_data_wide_features <- full_data_wide_raw %>%
  arrange(date) %>%
  mutate(SPRD_10YCMT_FEDFUNDS = GS10 - FEDFUNDS,
         D_UNRATE = UNRATE - lag(UNRATE, 12),
         G_CPIU = (CPIAUCSL / lag(CPIAUCSL, 12) - 1) * 100,
         D_EFFR = FEDFUNDS - lag(FEDFUNDS, 12),
         D_GS10 = GS10 - lag(GS10, 12)
         ) %>%
  mutate(across(
    .cols=c(SPRD_10YCMT_FEDFUNDS, D_UNRATE,
             G_CPIU, D_EFFR, GS10, D_GS10),
    .fns=list(lag1 = ~lag(.x, 1),
         lag3 = ~lag(.x, 3),
         lag6 = ~lag(.x, 6),
         lag9 = ~lag(.x, 9),
         lag12 = ~lag(.x, 12))
  )) %>%
  select(-CPIAUCSL) %>% ## index rises with time
  drop_na()
```

Calculate Adstock

The adstock transformation is an auto-regressive transformation of a time series. The transformation takes into account past values of the time series. The intuition is that past values of the time series has a contemporaneous effect on the outcome.

$$AdStock(x_t) = x_t + \theta AdStock(x_{t-1})$$

where $0 < \theta < 1$.

The parameters cannot be estimated easily with least squares or logistic regression. Instead, we assume a range of potential values between 0.05 and 0.85.

```
full_data_wide_features_adstock <- full_data_wide_features %>%
  arrange(date) %>%
    mutate(across(
    .cols=c(UNRATE:D_GS10),
    .fns=list(adstkL = ~stats::filter(.x,
                                  filter=0.05,
```

```
                                    method="recursive") ,
         adstkM = ~stats::filter(.x,
                                    filter=0.45,
                                    method="recursive") ,
         adstkHL = ~stats::filter(.x,
                                    filter=0.65,
                                    method="recursive"),
         adstkHM = ~stats::filter(.x,
                                    filter=0.75,
                                    method="recursive"),
         adstkHH = ~stats::filter(.x,
                                    filter=0.85,
                                    method="recursive")
  ))) %>%
  mutate(constant=1)
```

Recession in next 6 months

```
full_data_wide <- full_data_wide_features_adstock %>%
  arrange(date) %>%
  mutate(USREC_LEAD1 = lead(USREC, 1),
         USREC_LEAD2 = lead(USREC, 2),
         USREC_LEAD3 = lead(USREC, 3),
         USREC_LEAD4 = lead(USREC, 4),
         USREC_LEAD5 = lead(USREC, 5),
         USREC_LEAD6 = lead(USREC, 6),
         FUTREC = pmax(USREC_LEAD1, USREC_LEAD2, USREC_LEAD3,
                             USREC_LEAD4, USREC_LEAD5, USREC_LEAD6)) %>%
  drop_na() %>%
  select( -USREC_LEAD1, -USREC_LEAD2, -USREC_LEAD3,
         -USREC_LEAD4, -USREC_LEAD5, -USREC_LEAD6)
```

Split Train/Test

```
full_size <- nrow(full_data_wide)

train_size <- floor(full_size*0.80)

train_id <- seq.int(1,train_size,1)

full_data_wide$constant <- 1

train_data <- full_data_wide[train_id,]
test_data <- full_data_wide[-train_id,]


tbl_summary(train_data)
```

Characteristic	**N = 632**
date	1956-07-01 to 2009-02-01
USREC	97 (15%)
UNRATE	5.60 (4.90, 6.80)
GS10	6.19 (4.46, 7.92)
FEDFUNDS	5.2 (3.4, 7.4)
SPRD_10YCMT_FEDFUNDS	0.97 (0.11, 2.07)
D_UNRATE	-0.20 (-0.53, 0.40)
G_CPIU	3.27 (2.21, 4.89)
D_EFFR	0.07 (-1.14, 1.28)
D_GS10	0.10 (-0.64, 0.65)
SPRD_10YCMT_FEDFUNDS_lag1	0.96 (0.11, 2.06)
SPRD_10YCMT_FEDFUNDS_lag3	0.95 (0.11, 2.04)
SPRD_10YCMT_FEDFUNDS_lag6	0.93 (0.11, 2.03)
SPRD_10YCMT_FEDFUNDS_lag9	0.93 (0.11, 2.02)
SPRD_10YCMT_FEDFUNDS_lag12	0.93 (0.11, 2.02)
D_UNRATE_lag1	-0.20 (-0.53, 0.40)
D_UNRATE_lag3	-0.20 (-0.60, 0.40)
D_UNRATE_lag6	-0.20 (-0.60, 0.40)
D_UNRATE_lag9	-0.20 (-0.60, 0.40)
D_UNRATE_lag12	-0.20 (-0.60, 0.40)
G_CPIU_lag1	3.27 (2.21, 4.89)
G_CPIU_lag3	3.27 (2.21, 4.89)
G_CPIU_lag6	3.26 (2.17, 4.85)
G_CPIU_lag9	3.25 (2.14, 4.81)
G_CPIU_lag12	3.23 (2.09, 4.81)
D_EFFR_lag1	0.08 (-1.13, 1.28)
D_EFFR_lag3	0.10 (-1.10, 1.29)
D_EFFR_lag6	0.12 (-1.09, 1.29)
D_EFFR_lag9	0.13 (-1.02, 1.31)
D_EFFR_lag12	0.15 (-1.00, 1.31)
GS10_lag1	6.19 (4.46, 7.92)
GS10_lag3	6.19 (4.46, 7.92)
GS10_lag6	6.19 (4.46, 7.92)
GS10_lag9	6.19 (4.46, 7.92)
GS10_lag12	6.19 (4.46, 7.92)
D_GS10_lag1	0.11 (-0.64, 0.65)
D_GS10_lag3	0.11 (-0.63, 0.65)
D_GS10_lag6	0.13 (-0.62, 0.65)
D_GS10_lag9	0.15 (-0.61, 0.65)
D_GS10_lag12	0.16 (-0.59, 0.65)
UNRATE_adstkL	5.90 (5.15, 7.15)
UNRATE_adstkM	10.22 (8.85, 12.33)
UNRATE_adstkHL	16.0 (13.9, 19.3)
UNRATE_adstkHM	22.5 (19.6, 26.9)
UNRATE_adstkHH	38 (33, 44)
GS10_adstkL	6.51 (4.68, 8.34)
GS10_adstkM	11.2 (8.1, 14.4)
GS10_adstkHL	18 (13, 23)
GS10_adstkHM	25 (18, 32)
GS10_adstkHH	42 (29, 53)
FEDFUNDS_adstkL	5.5 (3.6, 7.8)
FEDFUNDS_adstkM	9.6 (6.2, 13.5)

Characteristic	N = 632
FEDFUNDS_adstkHL	15 (10, 21)
FEDFUNDS_adstkHM	21 (14, 30)
FEDFUNDS_adstkHH	35 (22, 50)
SPRD_10YCMT_FEDFUNDS_adstkL	1.02 (0.12, 2.18)
SPRD_10YCMT_FEDFUNDS_adstkM	1.75 (0.26, 3.65)
SPRD_10YCMT_FEDFUNDS_adstkHL	2.7 (0.4, 5.5)
SPRD_10YCMT_FEDFUNDS_adstkHM	4 (0, 7)
SPRD_10YCMT_FEDFUNDS_adstkHH	6 (1, 12)
D_UNRATE_adstkL	-0.21 (-0.56, 0.43)
D_UNRATE_adstkM	-0.37 (-1.01, 0.76)
D_UNRATE_adstkHL	-0.59 (-1.59, 1.26)
D_UNRATE_adstkHM	-0.9 (-2.1, 1.8)
D_UNRATE_adstkHH	-1.6 (-3.3, 3.1)
G_CPIU_adstkL	3.4 (2.3, 5.1)
G_CPIU_adstkM	6.0 (4.1, 8.9)
G_CPIU_adstkHL	9 (7, 14)
G_CPIU_adstkHM	13 (9, 19)
G_CPIU_adstkHH	21 (15, 32)
D_EFFR_adstkL	0.07 (-1.19, 1.34)
D_EFFR_adstkM	0.2 (-2.0, 2.3)
D_EFFR_adstkHL	0.4 (-3.1, 3.5)
D_EFFR_adstkHM	1 (-4, 5)
D_EFFR_adstkHH	1 (-7, 7)
D_GS10_adstkL	0.10 (-0.67, 0.68)
D_GS10_adstkM	0.19 (-1.17, 1.16)
D_GS10_adstkHL	0.27 (-1.74, 1.74)
D_GS10_adstkHM	0.4 (-2.2, 2.4)
D_GS10_adstkHH	0.5 (-3.0, 3.4)
constant	632 (100%)
FUTREC	143 (23%)

Automated Approaches

1. Gradient Boosting for Additive Models
2. eXtreme Gradient Boosting Trees
3. Random Forest
4. Stepwise Regression
5. Elastic Net (Lasso)
6. Multivariate Adaptive Regression Splines
7. Null Model: Intercept-only Model

```
fitControl <- trainControl(method = "timeslice",
                           initialWindow=392,
                           horizon=120,
                           fixedWindow=FALSE,
                           skip=119,
                           ## Estimate class probabilities
                           classProbs = TRUE,
                           ## Evaluate performance using
                           ## the following function
                           summaryFunction = mnLogLoss)
```

```r
train_yes_no <- train_data %>%
  mutate(FUTREC = case_when(FUTREC == 1 ~ "yes",
                            TRUE ~ "no"))

train_yes_no$FUTREC <- factor(train_yes_no$FUTREC,
                              levels=c("yes","no"))

train_yes_no_relevel <- train_yes_no

train_yes_no_relevel$FUTREC <- relevel(
  train_yes_no_relevel$FUTREC, ref="no"

)
```

```r
library(doParallel)

cl <- makePSOCKcluster(5)
registerDoParallel(cl)

set.seed(111)

gam_mod <- train(
  FUTREC ~ . - date - USREC - constant,
  data = train_yes_no,
  method = "gamboost",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  family = Binomial()
)

xgb_mod <- train(
  FUTREC ~ . - date - USREC - constant,
  data = train_yes_no,
  method = "xgbTree",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  objective  = "binary:logistic"
)

rf_mod <- train(
  FUTREC ~ . - date - USREC - constant,
  data = train_yes_no,
  method = "rf",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  importance = TRUE
)

stepwise_mod <- train(
  FUTREC ~ . - date - USREC - constant,
```

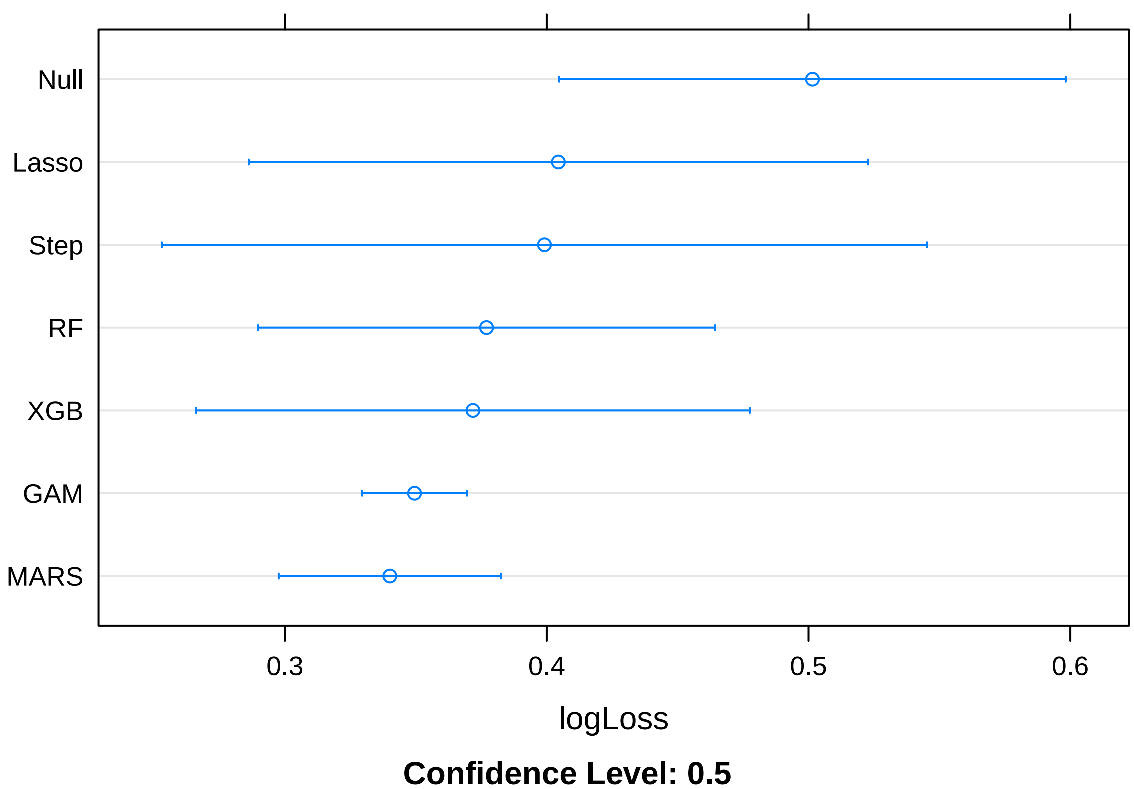
```
  data = train_yes_no_relevel,
  method = "glmStepAIC",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  family = binomial,
  trace = 0,
  k = 5 * log(nrow(train_yes_no)),
  direction = "forward"
)

glmnet_mod <- train(
  FUTREC ~ . - date - USREC - constant,
  data = train_yes_no,
  method = "glmnet",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  family = "binomial"
)

earth_mod <- train(
  FUTREC ~ . - date - USREC - constant,
  data = train_yes_no,
  method = "earth",
  trControl = fitControl,
  metric = "logLoss",
  tuneLength = 10,
  glm = list(family = binomial)
)

null_mod <- train(
  FUTREC ~ constant,
  data = train_yes_no,
  method = "glm",
  trControl = fitControl,
  metric = "logLoss",
  family = binomial
)




stopCluster(cl)
```

```
resamps <- resamples(list(XGB = xgb_mod,
                          GAM = gam_mod,
                          RF = rf_mod,
                          Step = stepwise_mod,
                          Lasso = glmnet_mod,
                          MARS = earth_mod,
                          Null = null_mod)
                     )
```

```r
summary(resamps)
```

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: XGB, GAM, RF, Step, Lasso, MARS, Null
## Number of resamples: 2
##
## logLoss
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## XGB   0.2660639 0.3189441 0.3718243 0.3718243 0.4247045 0.4775847    0
## GAM   0.3294957 0.3395017 0.3495076 0.3495076 0.3595135 0.3695194    0
## RF    0.2897389 0.3333727 0.3770065 0.3770065 0.4206402 0.4642740    0
## Step  0.2529652 0.3260483 0.3991314 0.3991314 0.4722144 0.5452975    0
## Lasso 0.2861799 0.3453152 0.4044506 0.4044506 0.4635859 0.5227212    0
## MARS  0.2976276 0.3188455 0.3400634 0.3400634 0.3612812 0.3824991    0
## Null  0.4047818 0.4531525 0.5015233 0.5015233 0.5498940 0.5982648    0
```

```r
dotplot(resamps, metric = "logLoss", conf.level=0.5)
```




```r
gam_mod$bestTune
```

```
##   mstop prune
## 7   350    no
```

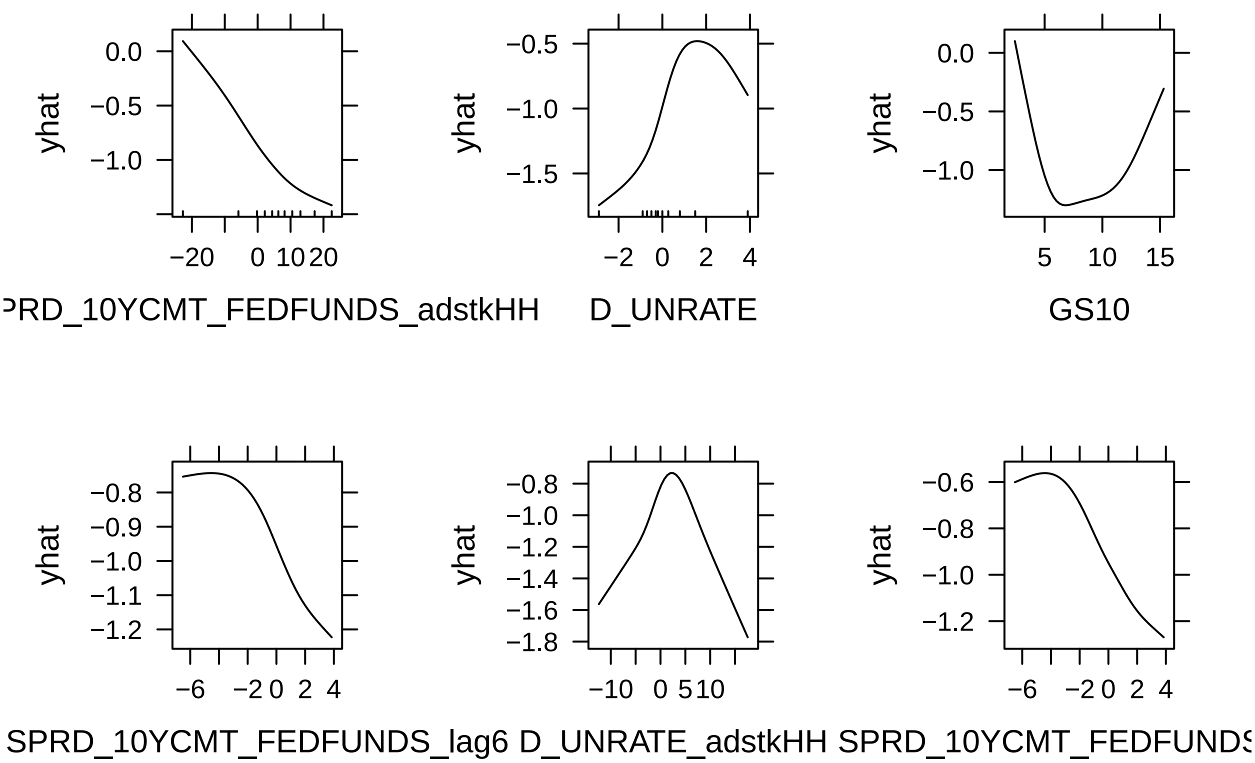
```r
df_imp <- as.data.frame(
  varimp(gam_mod$finalModel)) %>%
  select(variable, reduction) %>%
  arrange(desc(reduction))

df_imp$variable <- as.character(df_imp$variable)

knitr::kable(df_imp)
```

variable	reduction
SPRD_10YCMT_FEDFUNDS_adstkHH	0.1176291
D_UNRATE	0.0468893
GS10	0.0282598
SPRD_10YCMT_FEDFUNDS_lag6	0.0239265
D_UNRATE_adstkHH	0.0192148
SPRD_10YCMT_FEDFUNDS_lag3	0.0187787
D_UNRATE_lag6	0.0164118
SPRD_10YCMT_FEDFUNDS_lag12	0.0156730
G_CPIU_lag3	0.0126526
D_EFFR_adstkHL	0.0077577
G_CPIU_lag6	0.0063949
UNRATE_adstkHM	0.0058210
D_GS10_lag12	0.0053408
SPRD_10YCMT_FEDFUNDS_lag9	0.0049451
D_UNRATE_lag9	0.0039895
UNRATE_adstkHL	0.0036928
D_UNRATE_lag3	0.0036552
SPRD_10YCMT_FEDFUNDS_adstkL	0.0022550
G_CPIU	0.0018717
GS10_lag12	0.0017276
D_UNRATE_adstkM	0.0013765
SPRD_10YCMT_FEDFUNDS_adstkM	0.0010662
FEDFUNDS	0.0006654
D_EFFR	0.0003880
G_CPIU_lag1	0.0002735
D_EFFR_adstkM	0.0002094
UNRATE	0.0000000
SPRD_10YCMT_FEDFUNDS	0.0000000
D_GS10	0.0000000
SPRD_10YCMT_FEDFUNDS_lag1	0.0000000
D_UNRATE_lag1	0.0000000
D_UNRATE_lag12	0.0000000
G_CPIU_lag9	0.0000000
G_CPIU_lag12	0.0000000
D_EFFR_lag1	0.0000000
D_EFFR_lag3	0.0000000
D_EFFR_lag6	0.0000000
D_EFFR_lag9	0.0000000
D_EFFR_lag12	0.0000000
GS10_lag1	0.0000000
GS10_lag3	0.0000000
GS10_lag6	0.0000000

variable	reduction
GS10_lag9	0.0000000
D_GS10_lag1	0.0000000
D_GS10_lag3	0.0000000
D_GS10_lag6	0.0000000
D_GS10_lag9	0.0000000
UNRATE_adstkL	0.0000000
UNRATE_adstkM	0.0000000
UNRATE_adstkHH	0.0000000
GS10_adstkL	0.0000000
GS10_adstkM	0.0000000
GS10_adstkHL	0.0000000
GS10_adstkHM	0.0000000
GS10_adstkHH	0.0000000
FEDFUNDS_adstkL	0.0000000
FEDFUNDS_adstkM	0.0000000
FEDFUNDS_adstkHL	0.0000000
FEDFUNDS_adstkHM	0.0000000
FEDFUNDS_adstkHH	0.0000000
SPRD_10YCMT_FEDFUNDS_adstkHL	0.0000000
SPRD_10YCMT_FEDFUNDS_adstkHM	0.0000000
D_UNRATE_adstkL	0.0000000
D_UNRATE_adstkHL	0.0000000
D_UNRATE_adstkHM	0.0000000
G_CPIU_adstkL	0.0000000
G_CPIU_adstkM	0.0000000
G_CPIU_adstkHL	0.0000000
G_CPIU_adstkHM	0.0000000
G_CPIU_adstkHH	0.0000000
D_EFFR_adstkL	0.0000000
D_EFFR_adstkHM	0.0000000
D_EFFR_adstkHH	0.0000000
D_GS10_adstkL	0.0000000
D_GS10_adstkM	0.0000000
D_GS10_adstkHL	0.0000000
D_GS10_adstkHM	0.0000000
D_GS10_adstkHH	0.0000000

```
pdp.top1 <- partial(gam_mod,
          pred.var = df_imp$variable[1],
          plot = TRUE,
          rug = TRUE)

pdp.top2 <- partial(gam_mod,
          pred.var = df_imp$variable[2],
          plot = TRUE,
          rug = TRUE)

pdp.top3 <- partial(gam_mod,
    pred.var = df_imp$variable[3],
    plot = TRUE,
    chull = TRUE
```

```
  )

pdp.top4 <- partial(gam_mod,
    pred.var = df_imp$variable[4],
    plot = TRUE,
    chull = TRUE
  )

pdp.top5 <- partial(gam_mod,
    pred.var = df_imp$variable[5],
    plot = TRUE,
    chull = TRUE
  )

pdp.top6 <- partial(gam_mod,
    pred.var = df_imp$variable[6],
    plot = TRUE,
    chull = TRUE
  )

grid.arrange(pdp.top1, pdp.top2, pdp.top3,
             pdp.top4, pdp.top5, pdp.top6, ncol = 3)
```



Monotone Gradient Boosting (with peeking)

Peeking means we use the insights from the automated models to choose variables in subsequent models. This is technically cheating and causes the cross-validation errors to be artificially low. This is addressed in the test set which does not have peeking bias.

```
top_predictors <- head(df_imp$variable)

best_predictor <- head(top_predictors, 1)

top_fmla <- as.formula(paste0("FUTREC ~",
                              paste0(top_predictors,
                                     collapse=" + ")))

top1_fmla <- as.formula(paste0("FUTREC ~",
                               paste0(best_predictor,
                                      collapse=" + ")))

constraints <- c(-1, 1, -1, -1, 1, -1)

monotone_constraints <- paste0(top_predictors, " = ", constraints, collapse=", \n")

cat(monotone_constraints)
```

```
## SPRD_10YCMT_FEDFUNDS_adstkHH = -1,
## D_UNRATE = 1,
## GS10 = -1,
## SPRD_10YCMT_FEDFUNDS_lag6 = -1,
## D_UNRATE_adstkHH = 1,
## SPRD_10YCMT_FEDFUNDS_lag3 = -1
```

```
library(doParallel)

cl <- makePSOCKcluster(5)
registerDoParallel(cl)

myGrid <- expand.grid(n.trees=seq(5,100,5),
                      interaction.depth=1,
                      shrinkage=c(0.01, 0.025, 0.05, 0.1,
                                  0.2, 0.3, 0.4, 0.5, 0.6,
                                  0.7, 0.8, 0.85, 0.9, 0.95,
                                  0.99),
                      n.minobsinnode=c(5,10,20,40,50,60,
                                       70, 80, 90, 100))

gbm_mod_mono <- train(
  top_fmla,
  data = train_yes_no,
  method = "gbm",
  trControl = fitControl,
  metric = "logLoss",
  tuneGrid = myGrid,
  distribution = "bernoulli",
```

```r
  verbose=FALSE,
  var.monotone = constraints

)

stopCluster(cl)
```

```r
gbm_mod_mono$bestTune
```

```
##      n.trees interaction.depth shrinkage n.minobsinnode
## 2703      15                 1      0.95             60
```

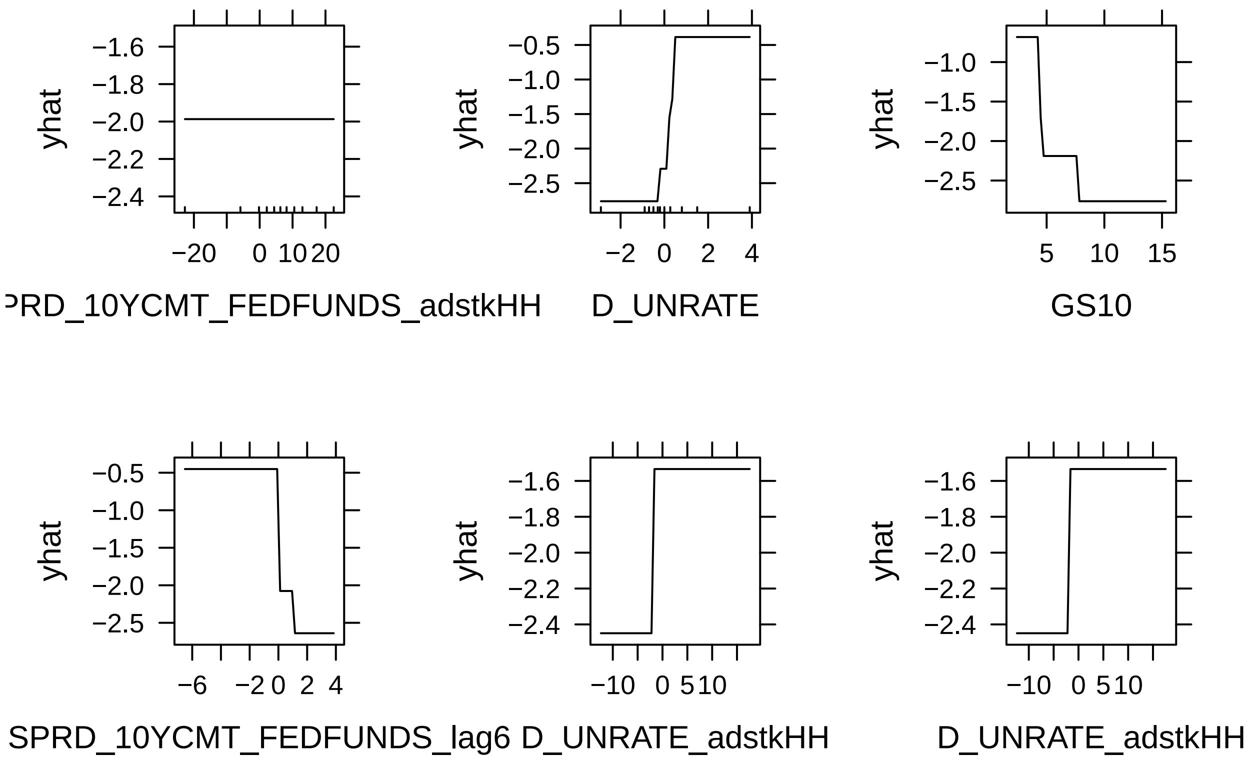
```r
varImp(gbm_mod_mono)
```

```
## gbm variable importance
##
##                              Overall
## SPRD_10YCMT_FEDFUNDS_lag6     100.00
## SPRD_10YCMT_FEDFUNDS_lag3      27.56
## D_UNRATE_adstkHH               23.76
## GS10                           19.85
## D_UNRATE                       15.72
## SPRD_10YCMT_FEDFUNDS_adstkHH    0.00
```

```r
pdp.top1 <- partial(gbm_mod_mono,
          pred.var = top_predictors[1],
          plot = TRUE,
          rug = TRUE)

pdp.top2 <- partial(gbm_mod_mono,
          pred.var =  top_predictors[2],
          plot = TRUE,
          rug = TRUE)

pdp.top3 <- partial(gbm_mod_mono,
    pred.var =  top_predictors[3],
    plot = TRUE,
    chull = TRUE
  )

pdp.top4 <- partial(gbm_mod_mono,
    pred.var =  top_predictors[4],
    plot = TRUE,
    chull = TRUE
  )

pdp.top5 <- partial(gbm_mod_mono,
    pred.var =  top_predictors[5],
    plot = TRUE,
    chull = TRUE
  )
```

```r
pdp.top6 <- partial(gbm_mod_mono,
    pred.var =  top_predictors[6],
    plot = TRUE,
    chull = TRUE
  )
```

```r
grid.arrange(pdp.top1, pdp.top2, pdp.top3,
             pdp.top4, pdp.top5, pdp.top5, ncol = 3)
```



One-Variable Logistic Regression (with peeking)

```r
library(doParallel)

cl <- makePSOCKcluster(5)
registerDoParallel(cl)

logit_mod <- train(
  top1_fmla,
  data = train_yes_no_relevel,
  method = "glm",
  trControl = fitControl,
```

```r
  metric = "logLoss",
  family=binomial
)

stopCluster(cl)

summary(logit_mod)
```

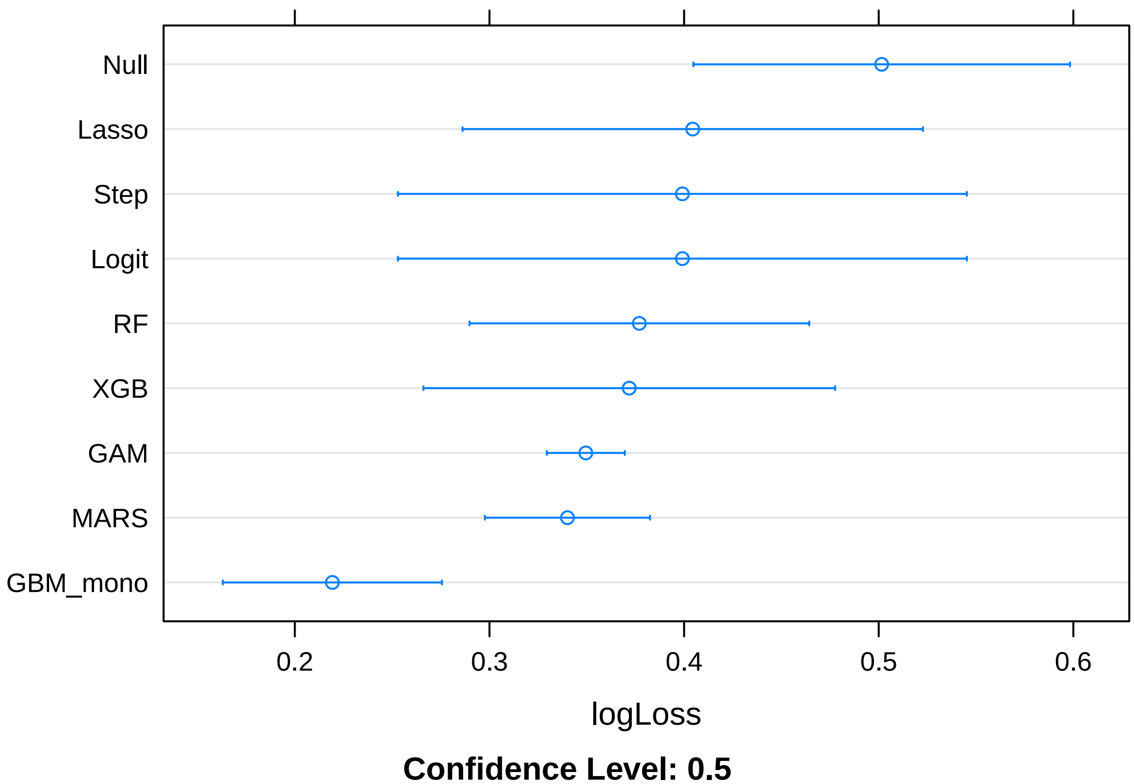
```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5532  -0.5774  -0.3226  -0.1154   2.6311
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -0.59442    0.12579  -4.726 2.29e-06 ***
## SPRD_10YCMT_FEDFUNDS_adstkHH -0.21457    0.01991 -10.777  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 675.89  on 631  degrees of freedom
## Residual deviance: 443.91  on 630  degrees of freedom
## AIC: 447.91
##
## Number of Fisher Scoring iterations: 6
```

```r
resamps <- resamples(list(XGB = xgb_mod,
                          GAM = gam_mod,
                          RF = rf_mod,
                          Step = stepwise_mod,
                          Lasso = glmnet_mod,
                          MARS = earth_mod,
                          GBM_mono = gbm_mod_mono,
                          Null = null_mod,
                          Logit = logit_mod)
                     )
summary(resamps)
```

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: XGB, GAM, RF, Step, Lasso, MARS, GBM_mono, Null, Logit
## Number of resamples: 2
##
## logLoss
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
```

```
## XGB      0.2660639 0.3189441 0.3718243 0.3718243 0.4247045 0.4775847    0
## GAM      0.3294957 0.3395017 0.3495076 0.3495076 0.3595135 0.3695194    0
## RF       0.2897389 0.3333727 0.3770065 0.3770065 0.4206402 0.4642740    0
## Step     0.2529652 0.3260483 0.3991314 0.3991314 0.4722144 0.5452975    0
## Lasso    0.2861799 0.3453152 0.4044506 0.4044506 0.4635859 0.5227212    0
## MARS     0.2976276 0.3188455 0.3400634 0.3400634 0.3612812 0.3824991    0
## GBM_mono 0.1630171 0.1911525 0.2192878 0.2192878 0.2474232 0.2755585    0
## Null     0.4047818 0.4531525 0.5015233 0.5015233 0.5498940 0.5982648    0
## Logit    0.2529652 0.3260483 0.3991314 0.3991314 0.4722144 0.5452975    0
```

```
dotplot(resamps, metric = "logLoss", conf.level=0.5)
```



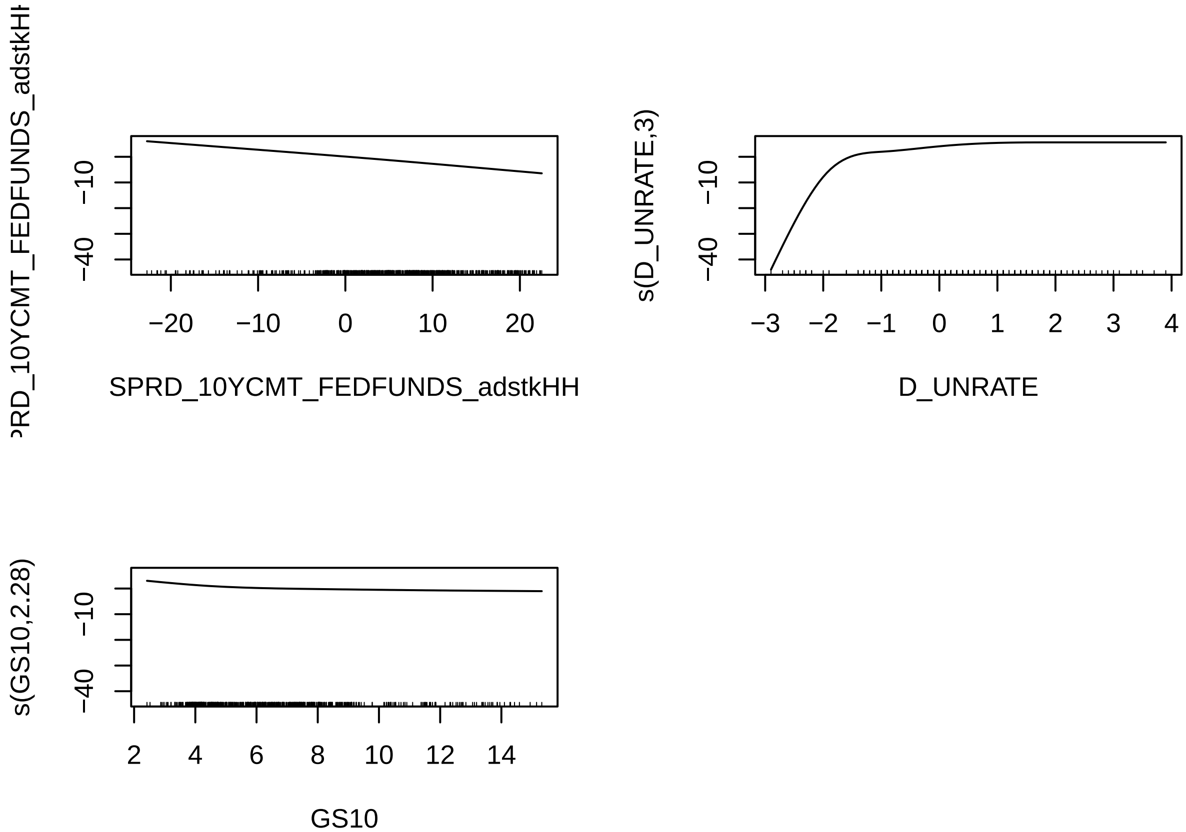
Shape-Constrained GAM (with peeking)

```
scam_mod <- scam(FUTREC ~ s(SPRD_10YCMT_FEDFUNDS_adstkHH, bs="mpd") +
                   s(D_UNRATE, bs="mpi") +
                   s(GS10, bs="mpd"),
                 data=train_data, family=binomial()
                 )

summary(scam_mod)
```

```
##
```

```
## Family: binomial
## Link function: logit
##
## Formula:
## FUTREC ~ s(SPRD_10YCMT_FEDFUNDS_adstkHH, bs = "mpd") + s(D_UNRATE,
##     bs = "mpi") + s(GS10, bs = "mpd")
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -81.81   30477.01  -0.003    0.998
##
## Approximate significance of smooth terms:
##                                  edf Ref.df Chi.sq  p-value
## s(SPRD_10YCMT_FEDFUNDS_adstkHH) 1.341  1.601  41.41 2.45e-09 ***
## s(D_UNRATE)                     3.002  3.000  21.66 7.69e-05 ***
## s(GS10)                         2.279  2.780  29.77 1.73e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.5064   Deviance explained = 48.4%
## UBRE score = -0.42421  Scale est. = 1         n = 632
##
## BFGS termination condition:
## 0.001605999
```

```
plot(scam_mod,pages=1,se=FALSE,
     all.terms=TRUE)
```



Logit with Knots (with peeking)

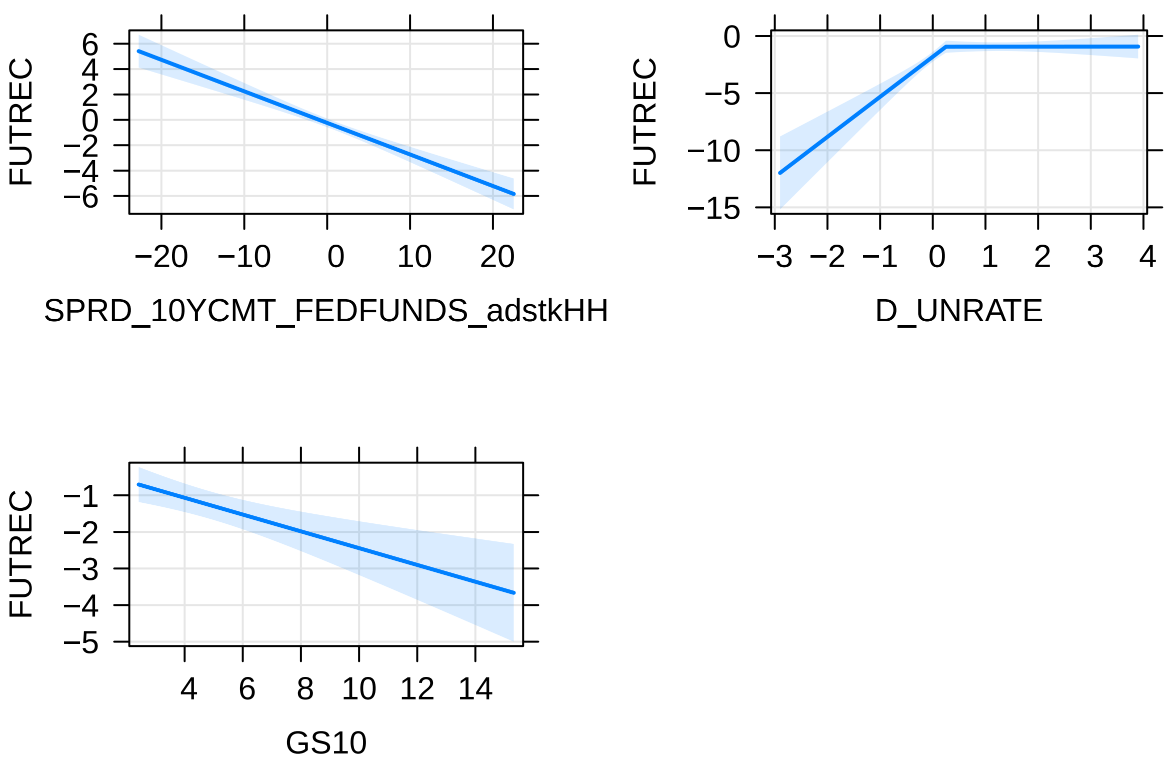
```r
logit_mod_knot <- glm(FUTREC ~ SPRD_10YCMT_FEDFUNDS_adstkHH +
                   D_UNRATE +
                   pmax(0,D_UNRATE - 0.25) +
                   GS10 ,
                   data=train_data, family=binomial)

summary(logit_mod_knot)
```

```
##
## Call:
## glm(formula = FUTREC ~ SPRD_10YCMT_FEDFUNDS_adstkHH + D_UNRATE +
##     pmax(0, D_UNRATE - 0.25) + GS10, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6757  -0.4682  -0.1543  -0.0093   3.2279
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    1.13482    0.42609   2.663 0.007737 **
## SPRD_10YCMT_FEDFUNDS_adstkHH -0.24872    0.02732  -9.105  < 2e-16 ***
```

```
## D_UNRATE                      3.50756    0.56351   6.224 4.83e-10 ***
## pmax(0, D_UNRATE - 0.25)     -3.50417    0.67238  -5.212 1.87e-07 ***
## GS10                         -0.22953    0.06242  -3.677 0.000236 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 675.89  on 631  degrees of freedom
## Residual deviance: 362.51  on 627  degrees of freedom
## AIC: 372.51
##
## Number of Fisher Scoring iterations: 7
```

Effect Plot for Knots

```r
plot(predictorEffects(logit_mod_knot, focal.levels=1000),
     main=NULL,
     axes = list(
       grid = TRUE,
       x = list(rug = FALSE),
       y = list(type = "link")
     ))
```



Performance Metrics

```
test_preds <- predict(logit_mod, newdata=test_data, type="prob")[,"yes"]
null_preds <- predict(null_mod, newdata=test_data, type="prob")[,"yes"]
knot_preds <- predict(logit_mod_knot, newdata=test_data, type="response")
scam_preds <- predict(scam_mod, newdata=test_data, type="response")
gam_preds <- predict(gam_mod, newdata=test_data, type="prob")[,"yes"]
gbm_mono_preds <- predict(gbm_mod_mono, newdata=test_data, type="prob")[,"yes"]
mars_preds <- predict(earth_mod, newdata=test_data, type="prob")[,"yes"]




perf <- function(lst_preds, f_metric=caTools::colAUC, metricname="ROC-AUC"){
  map_dfr(lst_preds, function(x){
  f_metric(x, test_data$FUTREC)
}) %>%
  pivot_longer(everything(), names_to="model", values_to=metricname) %>%
  knitr::kable()
}

myPreds <- list(logit_reg=test_preds, null_model=null_preds,
                knot_reg=knot_preds, scam_mod = scam_preds,
                gam_mod=gam_preds, gbm_mono=gbm_mono_preds,
                mars_mod=mars_preds)

perf(myPreds, caTools::colAUC, "ROC-AUC")
```

model	ROC-AUC
logit_reg	0.8081081
null_model	0.5000000
knot_reg	0.8959459
scam_mod	0.8770270
gam_mod	0.7182432
gbm_mono	0.8966216
mars_mod	0.7236486

```
perf(myPreds, MLmetrics::LogLoss, "LogLoss")
```

model	LogLoss
logit_reg	0.1865014
null_model	0.3343445
knot_reg	0.2172964
scam_mod	0.4832079
gam_mod	0.4913200
gbm_mono	0.4452028
mars_mod	2.6012907

Probability of Recession (Most Recent Month)

```r
curr_data <- tail(full_data_wide_features_adstock, 1)

curr_data$date
```

```
## [1] "2022-10-01"
```

```r
mods <- list(
  logistic_reg = logit_mod,
  scam_mod = scam_mod,
  knot_mod = logit_mod_knot,
  baseline = null_mod,
  gam_mod = gam_mod,
  gbm_mod_mono = gbm_mod_mono,
  mars_mod = earth_mod
)

score_fun <- function(mods, dat) {
  output <- map_dfc(.x = mods, .f = function(x) {
    if(class(x)[1] != "train"){
      predict(x, newdata = dat, type = "response")
    } else(
       predict(x, newdata = dat, type = "prob")[,"yes"]

    )

  }) %>%
    pivot_longer(everything(), names_to = "model",
                 values_to = "prob_rec")

  output$prob_rec <- scales::percent(output$prob_rec)

  return(output)
}


knitr::kable(score_fun(mods, curr_data))
```

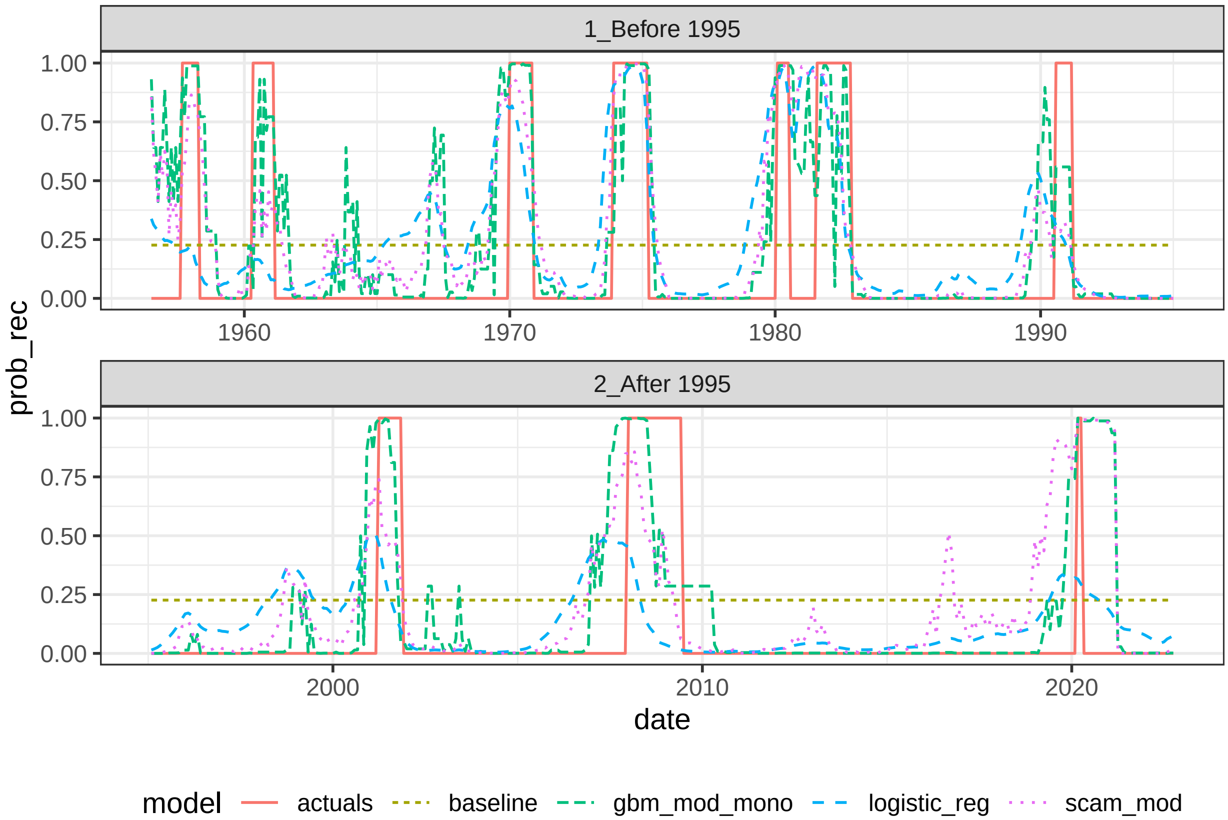
model	prob_rec
logistic_reg	7.53%
scam_mod	1.50%
knot_mod	0.57%
baseline	22.63%
gam_mod	40.15%
gbm_mod_mono	0.15%
mars_mod	4.37%

Backtesting

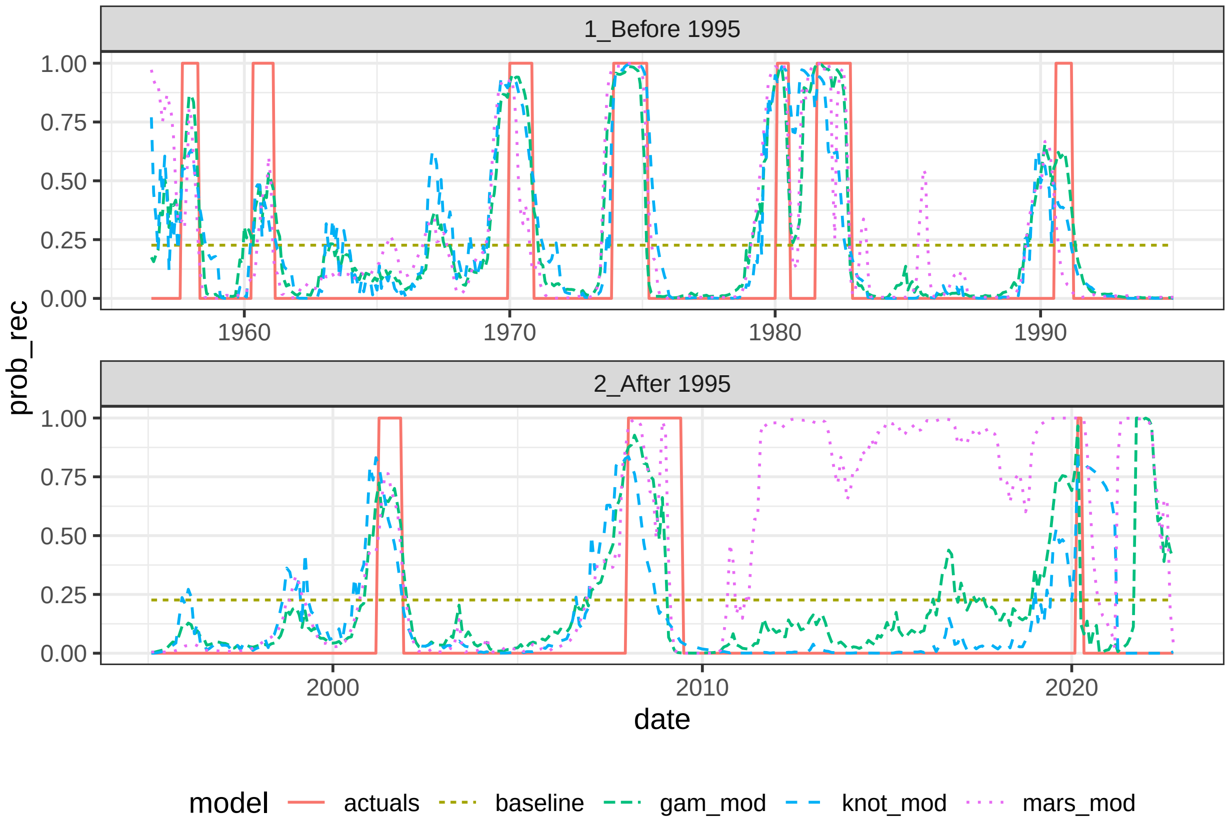
```
full_data_bktst <- full_data_wide_features_adstock

bkst_fun <- function(mods, dat) {
  output <- map_dfc(.x = mods, .f = function(x) {
    if(class(x)[1] != "train"){
      predict(x, newdata = dat, type = "response")
    } else(
       predict(x, newdata = dat, type = "prob")[,"yes"]

    )

  })

  output$date <- dat$date

  output <- output%>%
    pivot_longer(-date, names_to = "model",
                 values_to = "prob_rec")

  return(output)
}

df_plot <- bkst_fun(mods, full_data_bktst)

actuals <- full_data_bktst %>%
  mutate(model="actuals") %>%
  select(date, model, prob_rec=USREC)

df_plot_final <- bind_rows(df_plot, actuals)

df_plot_final <- df_plot_final %>%
  mutate(epoc = case_when(date <= "1995-01-01" ~ "1_Before 1995",
                          TRUE ~ "2_After 1995")
  )

df_plot_logit_scam <- df_plot_final %>%
  filter(model %in% c('actuals', 'baseline',
                      'logistic_reg',
                      'scam_mod', 'gbm_mod_mono'))

df_plot_knots_gbm <- df_plot_final %>%
  filter(model %in% c('actuals', 'baseline',
                      'knot_mod',
                      'gam_mod',
                      'mars_mod'))
```

```
ggplot(df_plot_logit_scam, aes(x=date, y=prob_rec, group=model,
                          linetype=model, color=model)) +
  geom_line() +
  theme_bw() +
  theme(legend.position = "bottom") +
```

```
  facet_wrap(vars(epoc), scales="free", nrow=2)
```




```
ggplot(df_plot_knots_gbm, aes(x=date, y=prob_rec, group=model,
                              linetype=model, color=model)) +
  geom_line() +
  theme_bw() +
  theme(legend.position = "bottom") +
  facet_wrap(vars(epoc), scales="free", nrow=2)
```



1_Before 1995
2_After 1995
prob_rec
1.00
0.75
0.50
0.25
0.00
1960
1970
1980
1990
2000
2010
2020
date
model
actuals
baseline
gam_mod
knot_mod
mars_mod