Application Exercises

Ken Chen

February 24, 2019

```r
setwd("D:/Perspectives/Computational Modeling/hw07")
set.seed(123)
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------------

## v ggplot2 3.1.0       v purrr   0.3.0
## v tibble  2.0.1       v dplyr   0.8.0.1
## v tidyr   0.8.2       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0

## -- Conflicts ---------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(readr)
library(ggplot2)
library(rsample)
library(margins)
library(splines)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
gss_train = read_csv("./data/gss_train.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   age = col_double(),
##   authoritarianism = col_double(),
##   childs = col_double(),
##   con_govt = col_double(),
##   egalit_scale = col_double(),
##   income06 = col_double(),
##   science_quiz = col_double(),
##   sibs = col_double(),
##   social_connect = col_double(),
##   tolerance = col_double(),
##   tvhours = col_double(),
##   wordsum = col_double()
## )

## See spec(...) for full column specifications.
```

```r
gss_test = read_csv("./data/gss_test.csv")
```

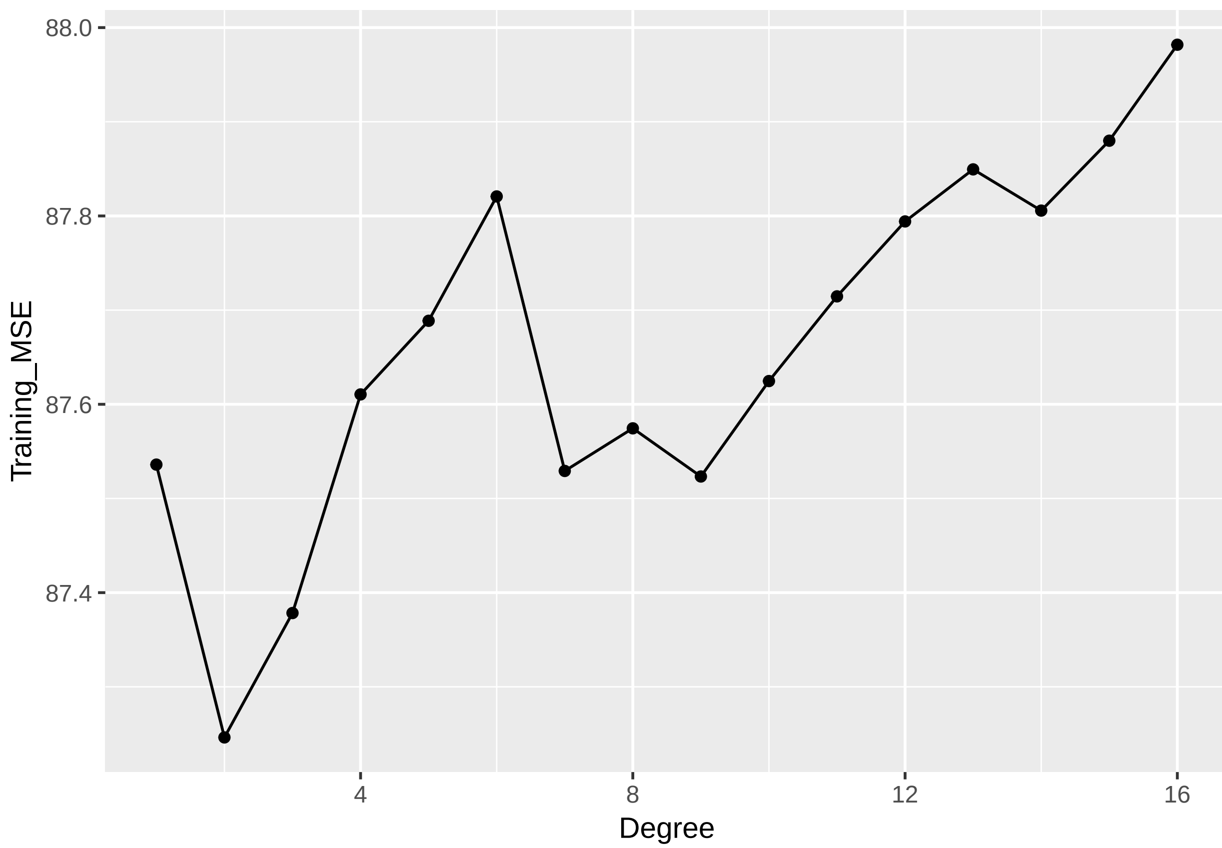
```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   age = col_double(),
##   authoritarianism = col_double(),
##   childs = col_double(),
##   con_govt = col_double(),
##   egalit_scale = col_double(),
##   income06 = col_double(),
##   science_quiz = col_double(),
##   sibs = col_double(),
##   social_connect = col_double(),
##   tolerance = col_double(),
##   tvhours = col_double(),
##   wordsum = col_double()
## )
## See spec(...) for full column specifications.
```

Application Exercises

Egalitarianism and income

1. Perform polynomial regression to predict egalit_scale using income06. Use 10-fold cross-validation to select the optimal degree d for the polynomial based on the MSE. Make a plot of the resulting polynomial fit to the data, and graph the average marginal effect (AME) of income06 across its potential values. Provide a substantive interpretation of the results.

```r
inc06 = select(.data = gss_train, income06)
ega = select(.data = gss_train, egalit_scale)
x_train = cbind(ega, inc06)

mse_lst = rep(0, 16)
cv = vfold_cv(data = x_train, v = 10)
for (i in 1:10){
  splited_set = cv$splits[[i]]
  train = analysis(splited_set); heldout = assessment(splited_set)
  y_true = heldout$egalit_scale
  for (j in 1:16){
    m = glm(egalit_scale ~ poly(income06, j, raw = TRUE), data = train)
    pred = predict(m, newdata = heldout)
    mse = sum((pred - y_true)^2)/length(y_true)
    mse_lst[j] = mse_lst[j] + mse
  }
}

mse_lst = mse_lst/10
tibble_poly = tibble(Training_MSE = mse_lst, Degree = 1:16)
tibble_poly %>%
  ggplot(aes(x = Degree, y = Training_MSE)) + geom_point() + geom_line()
```

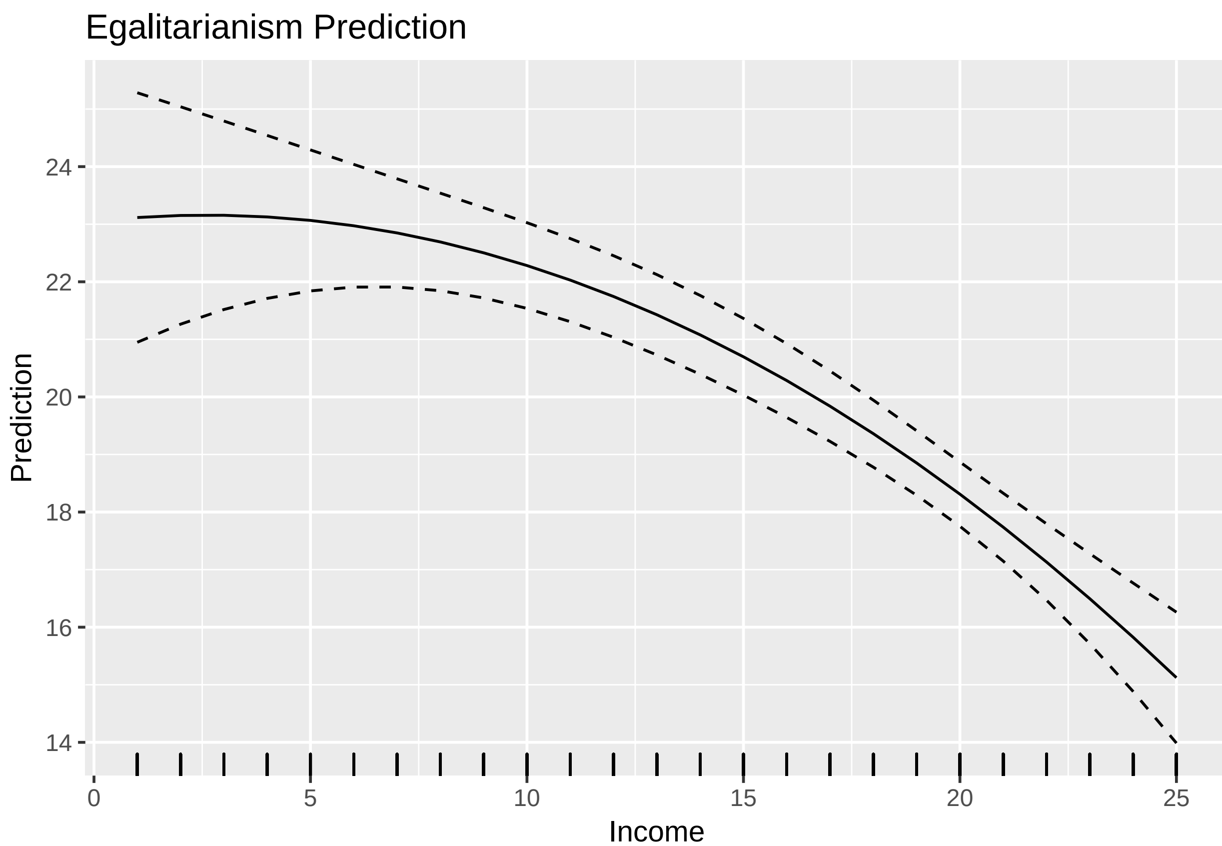


As we can see from the plot, the optimal degree is 2.

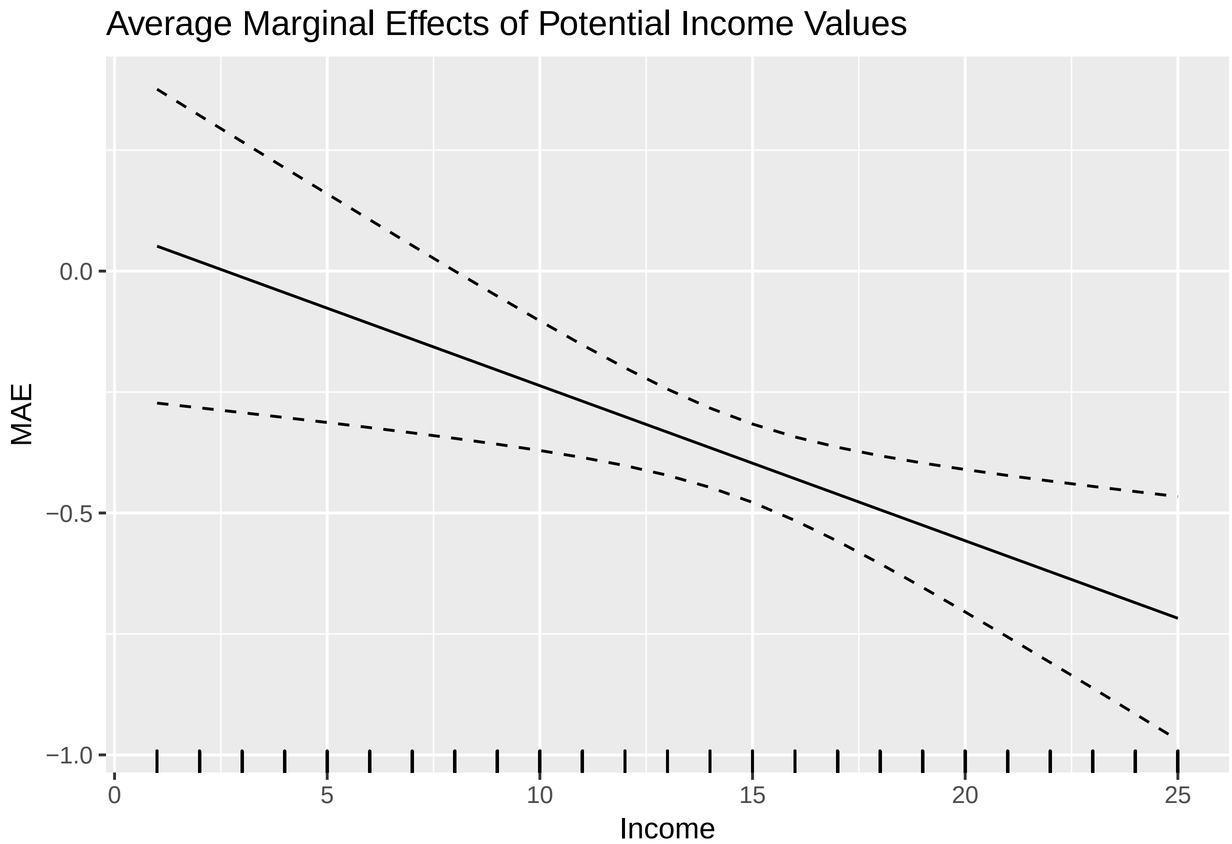
```r
poly_m = lm(egalit_scale ~ income06 + I(income06^2), data = gss_train)

cplot(poly_m, "income06", what='prediction', draw = F) %>%
  ggplot(aes(x = xvals)) +
  geom_line(aes(y = yvals)) +
  geom_line(aes(y = upper), linetype = 2) +
  geom_line(aes(y = lower), linetype = 2) +
  geom_rug(data = gss_train, aes(x = income06)) +
  labs(title = "Egalitarianism Prediction", x = 'Income', y = "Prediction")
```

```
##    xvals    yvals    upper    lower
## 1      1 23.11631 25.28371 20.94890
## 2      2 23.15180 25.04001 21.26359
## 3      3 23.15524 24.79232 21.51817
## 4      4 23.12665 24.54195 21.71134
## 5      5 23.06600 24.29036 21.84165
## 6      6 22.97331 24.03899 21.90764
## 7      7 22.84858 23.78870 21.90846
## 8      8 22.69180 23.53894 21.84467
## 9      9 22.50298 23.28678 21.71917
## 10    10 22.28211 23.02670 21.53752
## 11    11 22.02920 22.75132 21.30708
## 12    12 21.74424 22.45297 21.03552
## 13    13 21.42724 22.12502 20.72946
## 14    14 21.07819 21.76262 20.39376
```

```
## 15     15 20.69710 21.36290 20.03130
## 16     16 20.28396 20.92501 19.64291
## 17     17 19.83878 20.45044 19.22712
## 18     18 19.36155 19.94356 18.77955
## 19     19 18.85228 19.41247 18.29210
## 20     20 18.31096 18.86919 17.75274
```

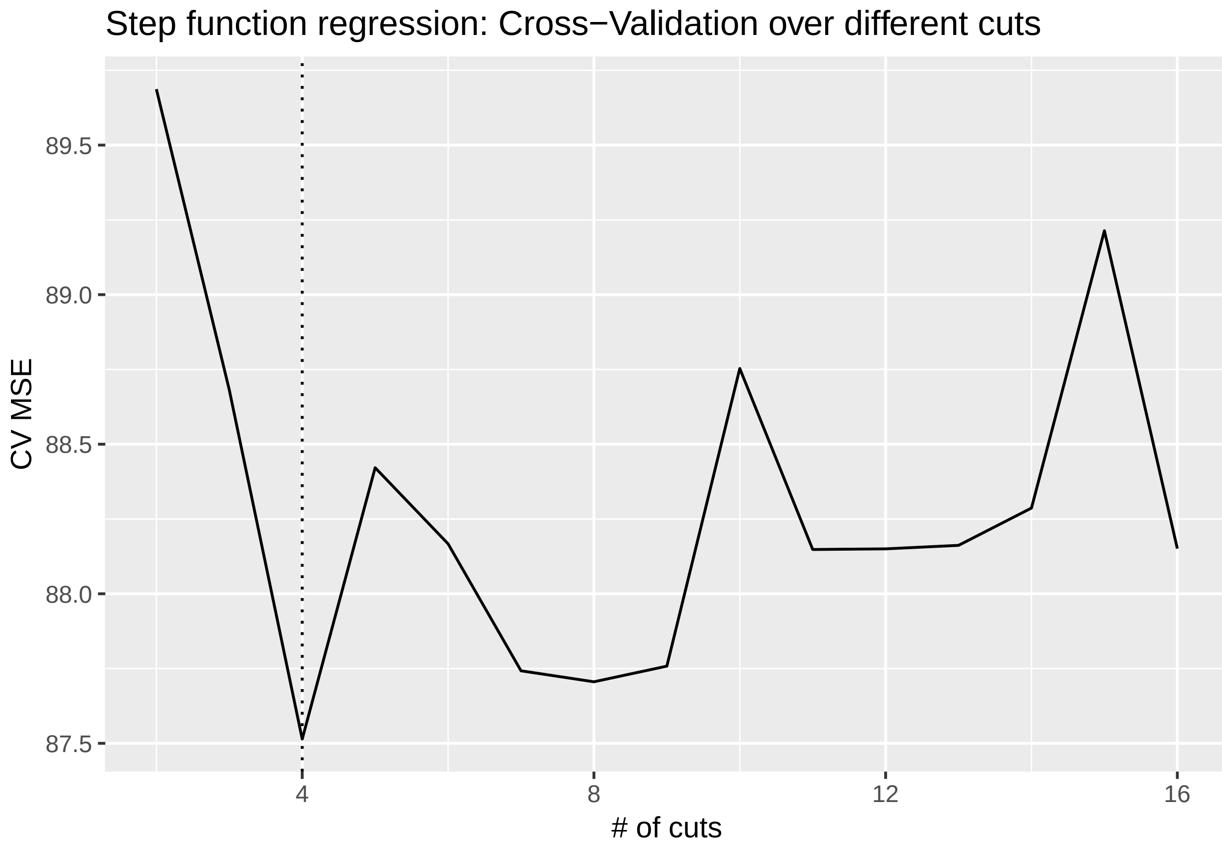



```r
cplot(poly_m, "income06", what='effect', draw = F) %>%
  ggplot(aes(x = xvals)) +
  geom_line(aes(y = yvals)) +
  geom_line(aes(y = upper), linetype = 2) +
  geom_line(aes(y = lower), linetype = 2) +
  geom_rug(data = gss_train, aes(x = income06)) +
  labs(title = "Average Marginal Effects of Potential Income Values", x = 'Income', y = "MAE")
```



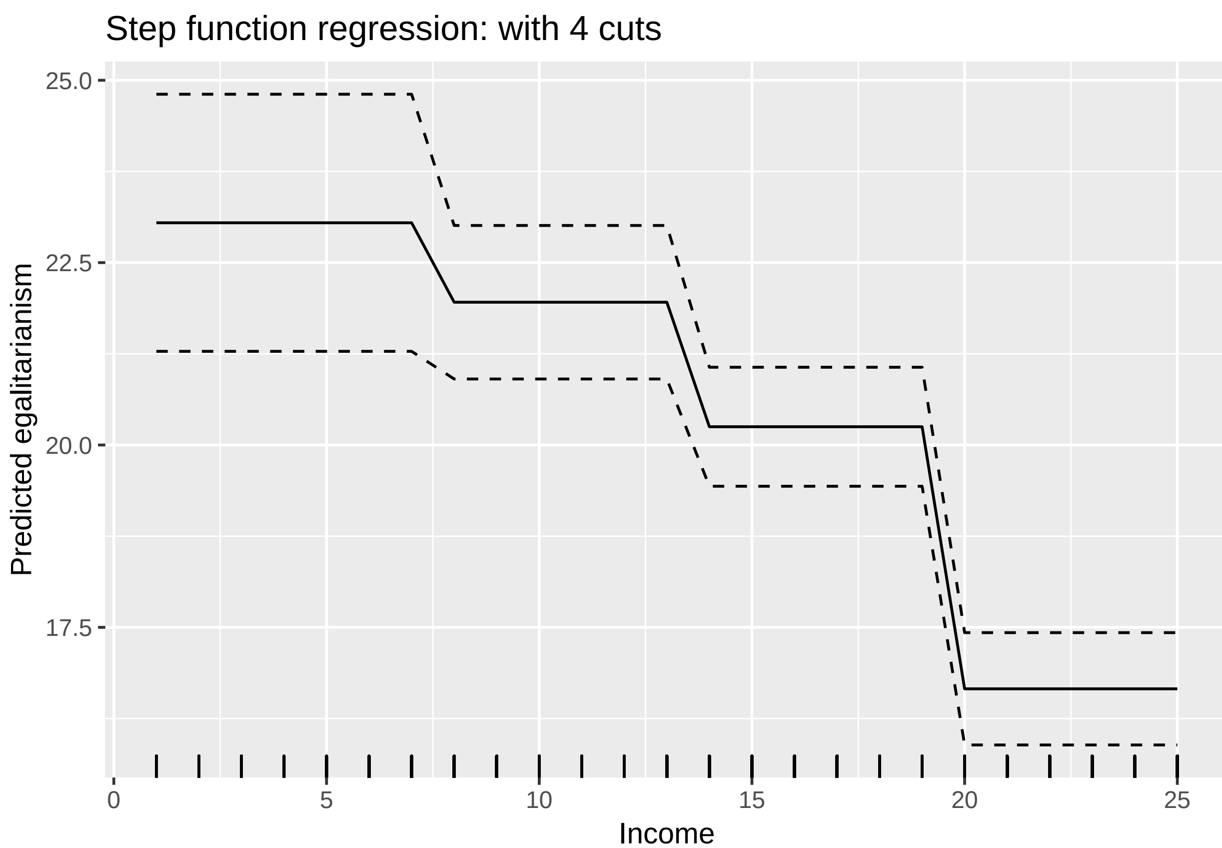
2. Fit a step function to predict egalit_scale using income06, and perform 10-fold cross-validation to choose the optimal number of cuts. Make a plot of the fit obtained and interpret the results.

```
mse_lst = rep(0, 15)
for (i in 2:16) {
  gss_train$inc06_cut = cut_interval(gss_train$income06, i)
  m = glm(egalit_scale ~ inc06_cut, data = gss_train)
  mse_lst[i-1] = boot::cv.glm(gss_train, m, K = 10)$delta[1]
}
tibble(cut_num = 2:16, mse = mse_lst) %>%
  ggplot(aes(cut_num, mse)) +
  geom_line() +
  geom_vline(xintercept = which.min(mse_lst) + 1, linetype = 3) +
  labs(title = "Step function regression: Cross-Validation over different cuts", x = "# of cuts", y = "(
```



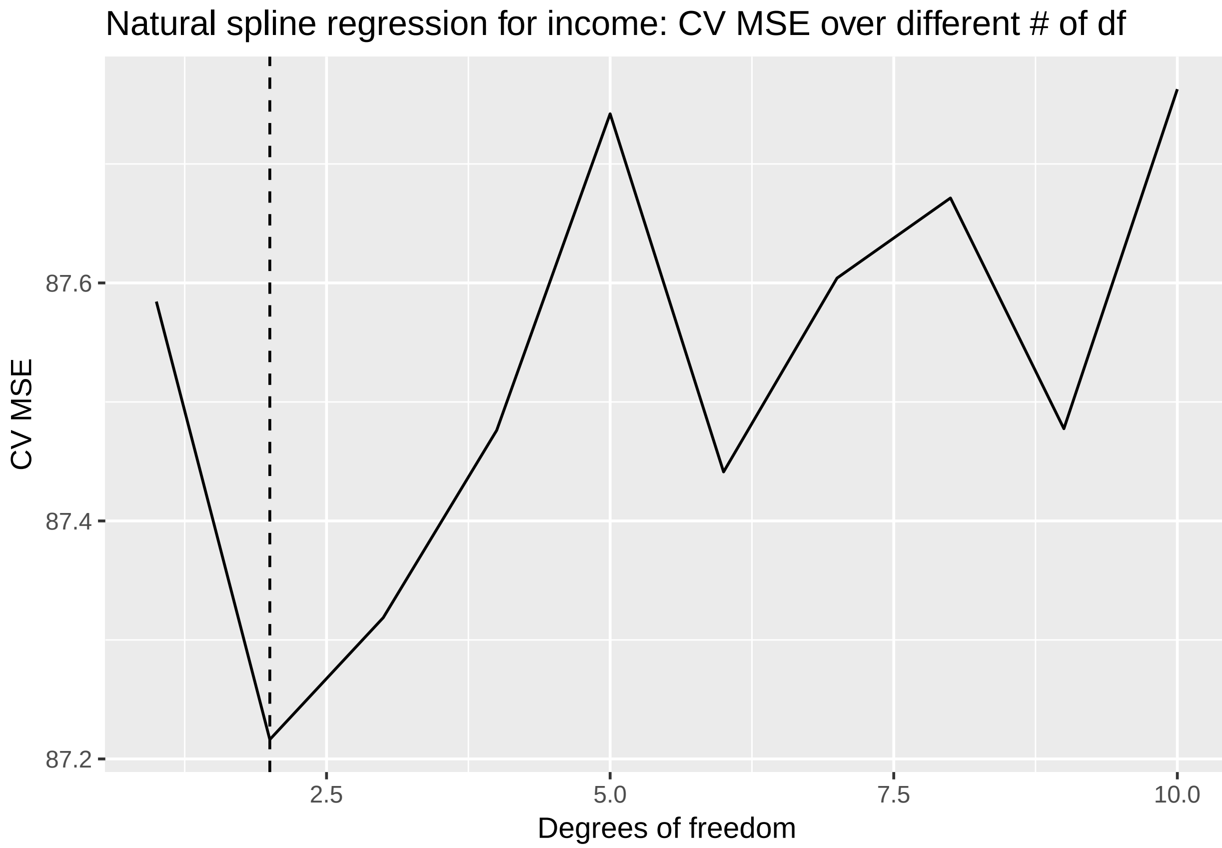
The optimal number of cuts is 4 according to our cross validation.

```
m_opt = lm(egalit_scale ~ cut_interval(income06, 4), data = gss_train)

m_opt %>% prediction %>%
  ggplot(aes(x = income06)) +
  geom_line(aes(y = fitted)) +
  geom_line(aes(y = fitted + 1.96 * se.fitted), linetype = 2) +
  geom_line(aes(y = fitted - 1.96 * se.fitted), linetype = 2) +
  geom_rug(data = gss_train, aes(x = income06)) +
  labs(title = "Step function regression: with 4 cuts", x = "Income", y = "Predicted egalitarianism")
```



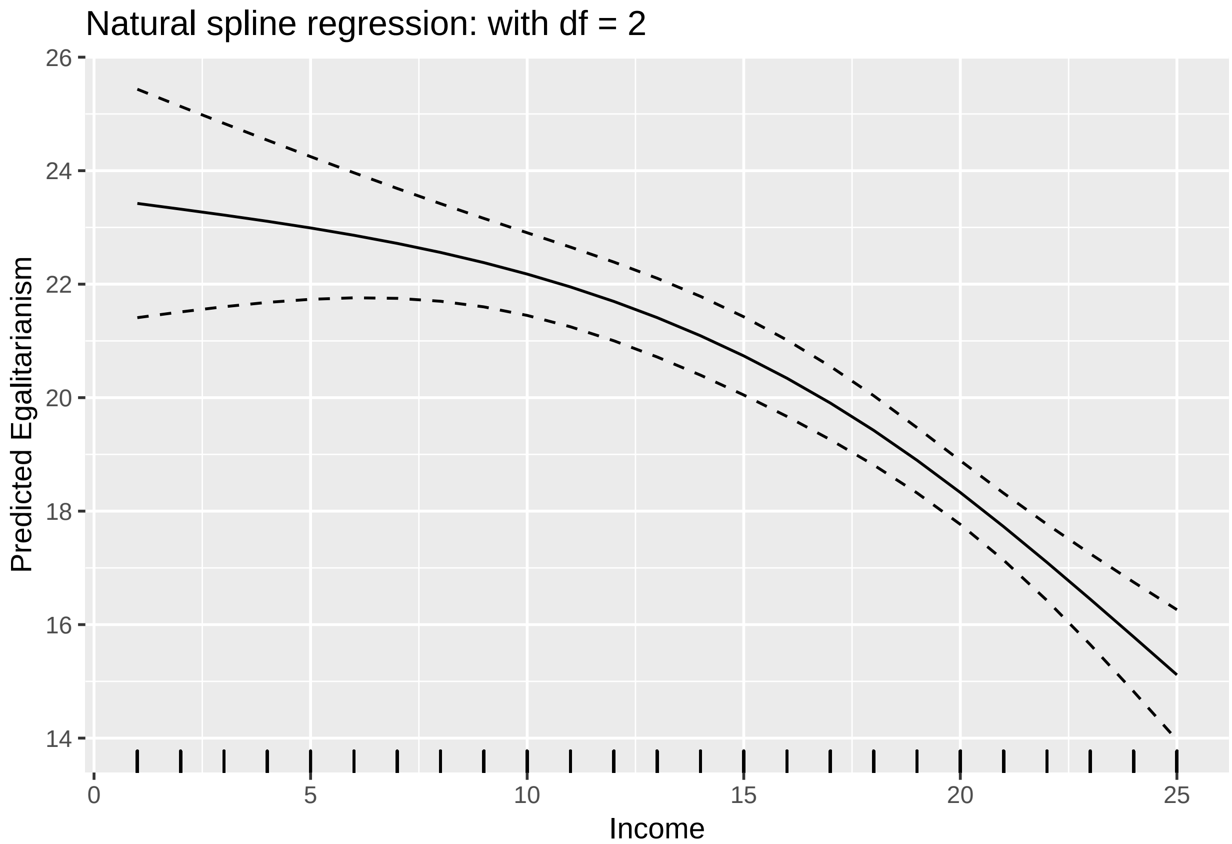
3.Fit a natural regression spline to predict egalit_scale using income06. Use 10-fold cross-validation to select the optimal number of degrees of freedom, and present the results of the optimal model.

```
mse_lst = rep(0, 10)
for (i in 1:10) {
  m_spline = glm(egalit_scale ~ ns(income06, df = i), data = gss_train)
  mse_lst[i] = boot::cv.glm(gss_train, m_spline, K = 10)$delta[1]
}
tibble(df = 1:10, mse = mse_lst) %>%
  ggplot(aes(df, mse)) +
  geom_line() +
  geom_vline(xintercept = which.min(mse_lst), linetype = 2) +
  labs(title = "Natural spline regression for income: CV MSE over different # of df", x = "Degrees of f
```



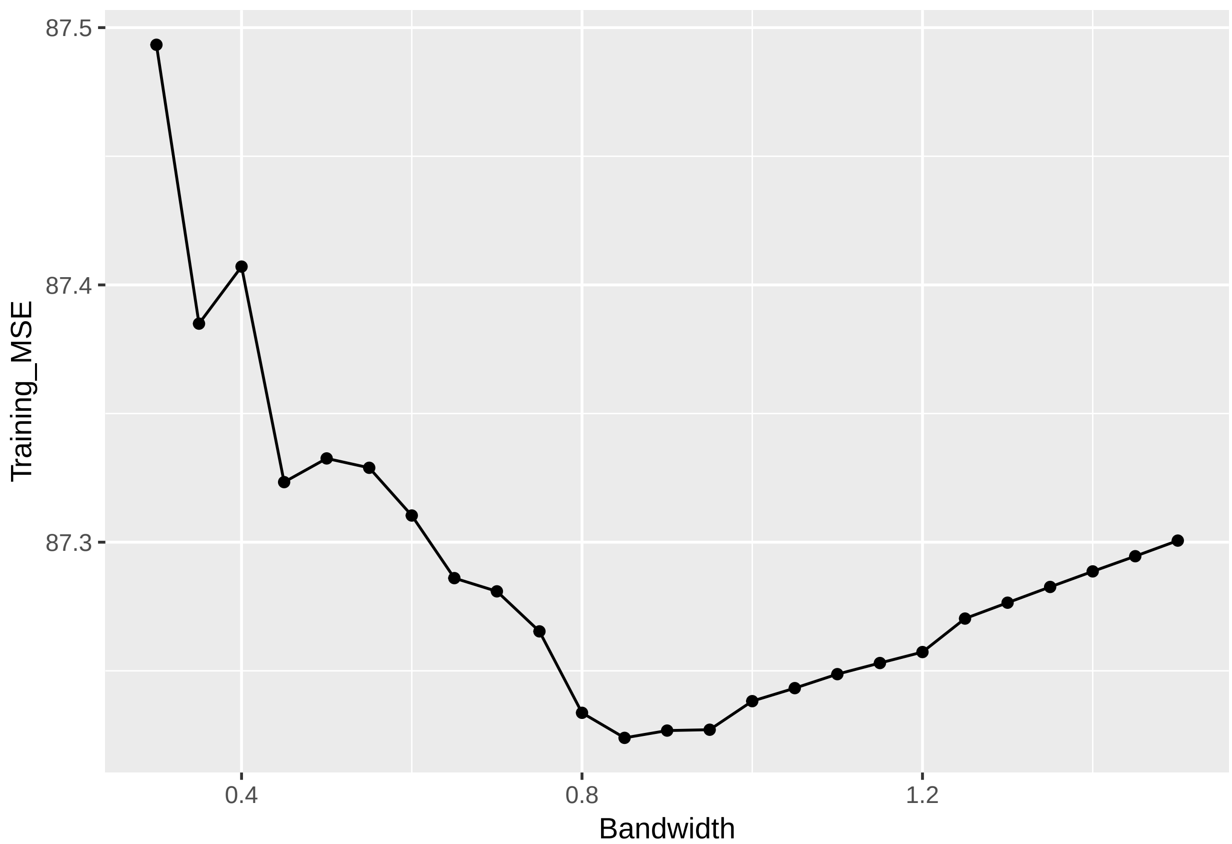
According to the 10-fold cross validation, the optimal degrees of freedom is 2.

```
ns_opt = lm(egalit_scale ~ ns(income06, df = 2), data = gss_train)

ns_opt %>% prediction %>%
  ggplot(aes(x = income06)) +
  geom_line(aes(y = fitted)) +
  geom_line(aes(y = fitted + 1.96 * se.fitted), linetype = 2) +
  geom_line(aes(y = fitted - 1.96 * se.fitted), linetype = 2) +
  geom_rug(data = gss_train, aes(x = income06)) +
  labs(title = "Natural spline regression: with df = 2", x = "Income", y = "Predicted Egalitarianism")
```



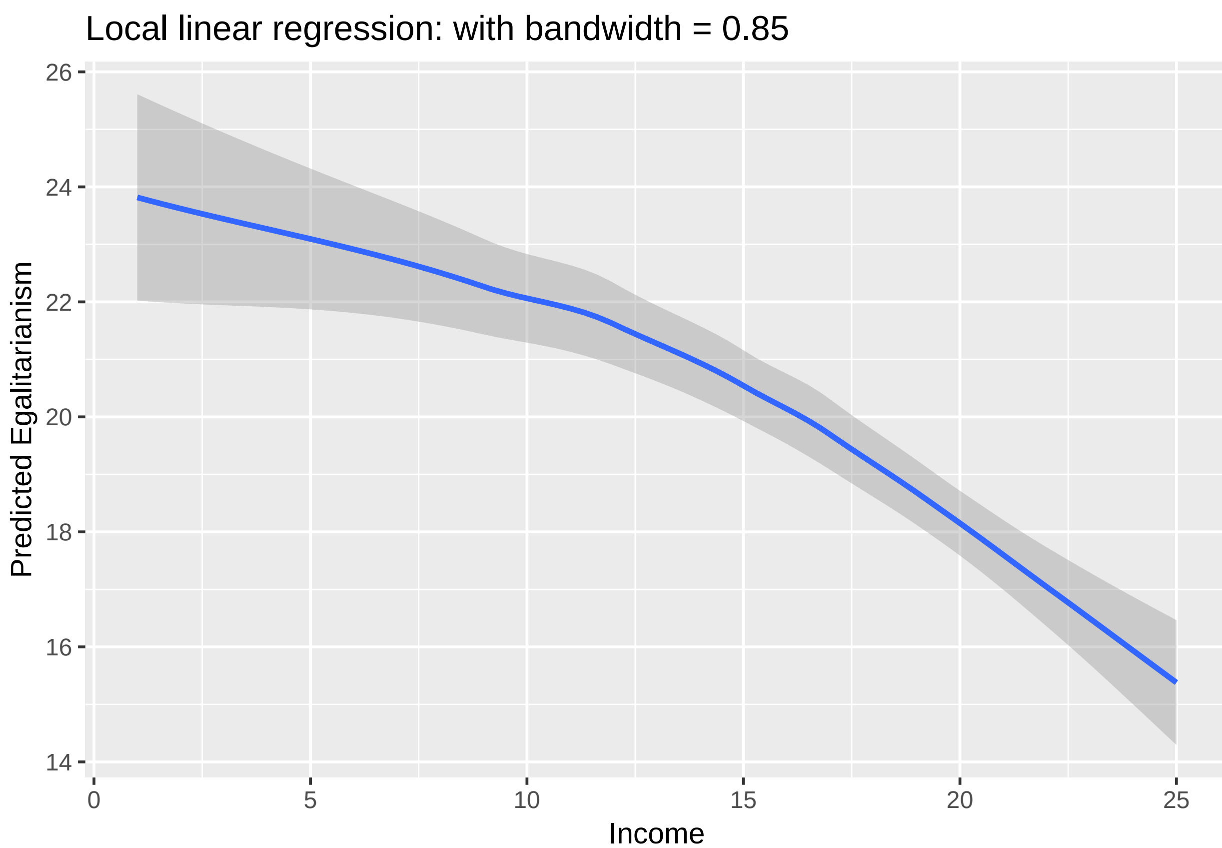
4. Fit a local linear regression model to predict egalit_scale using income06. Use 10-fold cross-validation to select the optimal bandwidth. Interpret the results.

```
mse_lst = rep(0, 25)
cv = vfold_cv(data = x_train, v = 10)
for (i in 1:10){
  splited_set = cv$splits[[i]]
  train = analysis(splited_set); heldout = assessment(splited_set)
  y_true = heldout$egalit_scale
  j = 1
  for (bdw in seq(0.3, 1.5, 0.05)){
    m = loess(egalit_scale ~ income06, data = train, span = bdw, degree = 1)
    pred = predict(m, newdata = heldout)
    mse = sum((pred - y_true)^2)/length(y_true)
    mse_lst[j] = mse_lst[j] + mse
    j = j+1
  }
}

mse_lst = mse_lst/10
tibble_poly = tibble(Training_MSE = mse_lst, Bandwidth = seq(0.3, 1.5, 0.05))
tibble_poly %>%
  ggplot(aes(x = Bandwidth, y = Training_MSE)) +
  geom_point() +
  geom_line()
```



The optimal bandwidth is 0.85.

```
ggplot(gss_train, aes(income06, egalit_scale)) +
  geom_smooth(method = "loess", span = 0.85, method.args = list(degree = 1)) +
  labs(title = "Local linear regression: with bandwidth = 0.85", x = "Income", y = "Predicted Egalitari
```

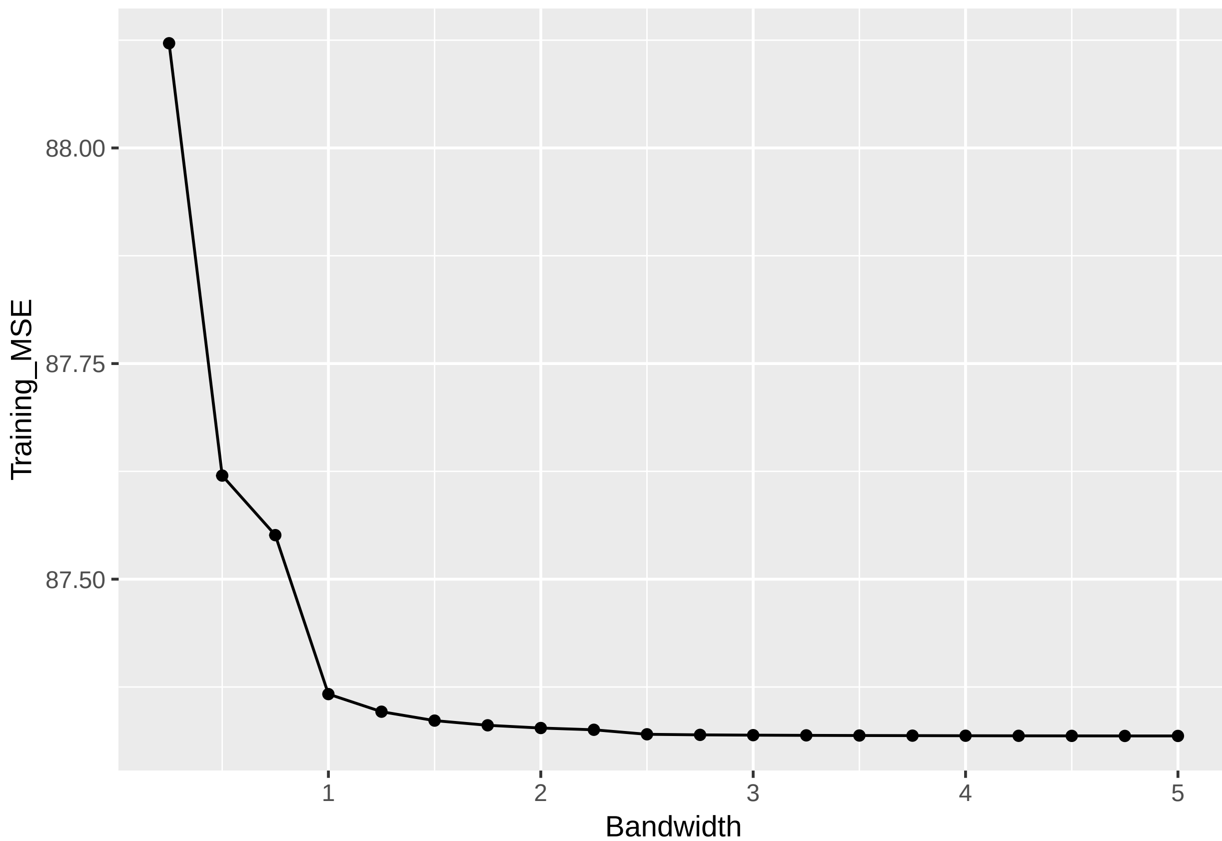


5. Fit a local polynomial regression model to predict egalit_scale using income06. Use 10-fold crossvalidation to select the optimal bandwidth. Interpret the results.

```
mse_lst = rep(0, 20)
cv = vfold_cv(data = x_train, v = 10)
for (i in 1:10){
  splited_set = cv$splits[[i]]
  train = analysis(splited_set); heldout = assessment(splited_set)
  y_true = heldout$egalit_scale
  j = 1
  for (bdw in seq(0.25, 5, 0.25)){
    m = loess(egalit_scale ~ income06, data = train, span = bdw, degree = 2)
    pred = predict(m, newdata = heldout)
    mse = sum((pred - y_true)^2)/length(y_true)
    mse_lst[j] = mse_lst[j] + mse
    j = j+1
  }
}
```

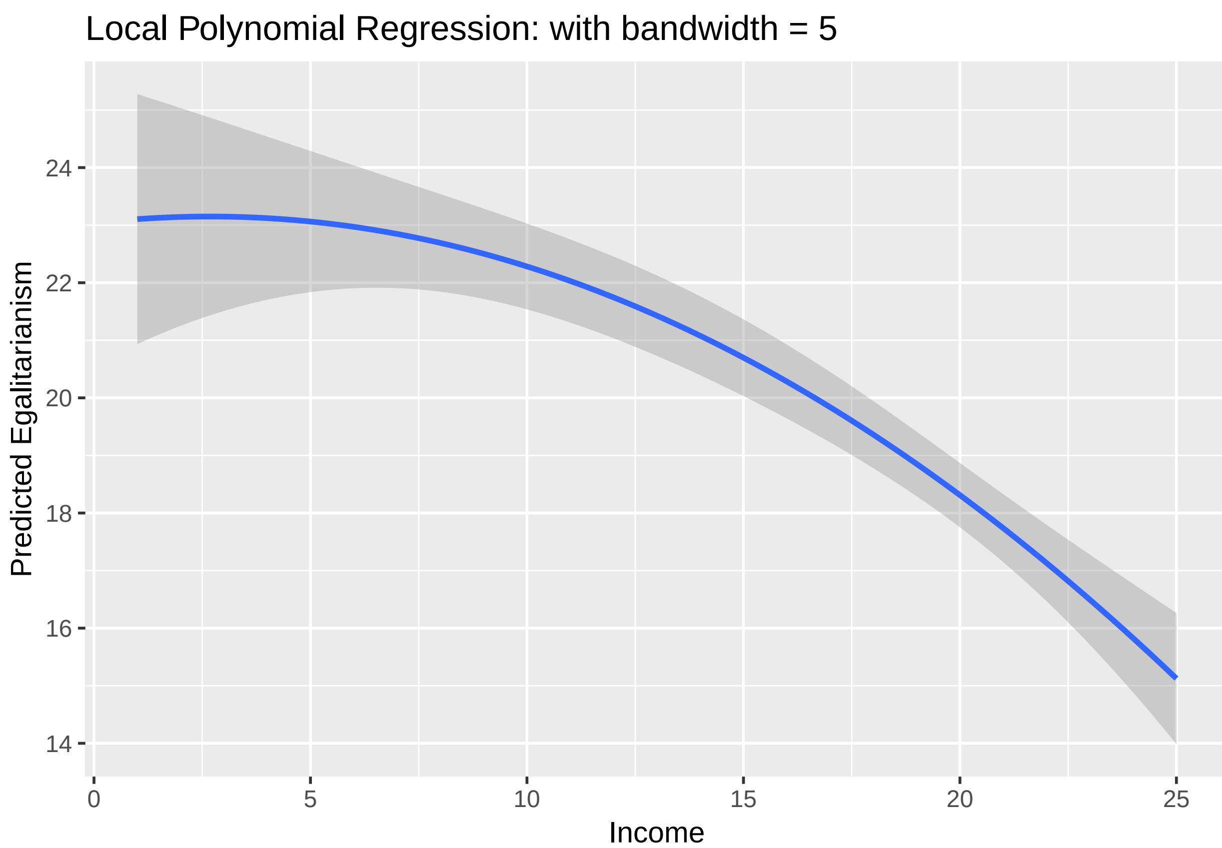
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 20

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```r
mse_lst = mse_lst/10
tibble_poly = tibble(Training_MSE = mse_lst, Bandwidth = seq(0.25, 5, 0.25))
tibble_poly %>%
  ggplot(aes(x = Bandwidth, y = Training_MSE)) +
  geom_point() +
  geom_line()
```



The optimal bandwidth is 5.

```r
ggplot(gss_train, aes(income06, egalit_scale)) +
  geom_smooth(method = "loess", span = 5, method.args = list(degree = 2)) +
  labs(title = "Local Polynomial Regression: with bandwidth = 5", x = "Income", y = "Predicted Egalitar
```



Egalitarianism and everything

```r
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand

## Loading required package: foreach

##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when

## Loaded glmnet 2.0-16
```

```r
library(pls)
```

```
##
## Attaching package: 'pls'

## The following object is masked from 'package:caret':
```

```
##
##     R2
## The following object is masked from 'package:stats':
##
##     loadings
```

```r
library(earth)
```

```
## Loading required package: plotmo
## Loading required package: plotrix
## Loading required package: TeachingDemos
```

```r
library(iml)
```

1. Estimate the following models using all the available predictors:

a. Linear regression
b. Elastic net regression
c. Principal component regression
d. Partial least squares regression
e. Multivariate adaptive regression splines (MARS)

- Perform appropriate data pre-processing (e.g. standardization) and hyperparameter tuning (e.g. lambda for PCR/PLS, lambda and alpha for elastic net, degree of interactions and number of retained terms for MARS)
- Use 10-fold cross-validation for each model to estimate the model's performance using MSE.

```r
gss_train = select(gss_train, -inc06_cut)
# Linear Regression Model
lr <- train(egalit_scale ~ .,data = gss_train,
  method = "lm", metric = "RMSE", trControl = trainControl(method = "cv", number = 10), preProcess = c(
)
# Elastic Net Regression
ela.net <- train( egalit_scale ~ ., data = gss_train, method = "glmnet",
  trControl = trainControl(method = "cv", number = 10), metric = "RMSE", preProcess = c("zv", "center",
)
# PCR
pcr <- train(egalit_scale ~ ., data = gss_train, method = "pcr",
  trControl = trainControl(method = "cv", number = 10), metric = "RMSE", preProcess = c("zv", "center",
)
# PLS
pls <- train(egalit_scale ~ ., data = gss_train, method = "pls",
  trControl = trainControl(method = "cv", number = 10), metric = "RMSE", preProcess = c("zv", "center",
)
# MARS
grid <- expand.grid(degree = 1:3, nprune = seq(2, 100, length.out = 10) %>% floor())
mars <- train(egalit_scale ~ ., data = gss_train, method = "earth",
  trControl = trainControl(method = "cv", number = 10), metric = "RMSE", preProcess = c("zv"), tuneGrid
)

summary(resamples(list(
  Linear.Regression = lr,
  Elastic.Net = ela.net,
  PCR = pcr,
  PLS = pls,
```

```r
  MARS = mars
)))
```

```
##
## Call:
## summary.resamples(object = resamples(list(Linear.Regression =
##  lr, Elastic.Net = ela.net, PCR = pcr, PLS = pls, MARS = mars)))
##
## Models: Linear.Regression, Elastic.Net, PCR, PLS, MARS
## Number of resamples: 10
##
## MAE
##                       Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## Linear.Regression 5.502586 6.014391 6.400598 6.253367 6.566769 6.725281
## Elastic.Net       5.766522 5.949770 6.150565 6.159137 6.385637 6.580547
## PCR               5.899741 6.268928 6.344180 6.449353 6.672020 7.076248
## PLS               5.647323 6.092515 6.282312 6.286030 6.389732 6.999243
## MARS              5.743503 5.914691 6.223570 6.171208 6.461125 6.539609
##                   NA's
## Linear.Regression    0
## Elastic.Net          0
## PCR                  0
## PLS                  0
## MARS                 0
##
## RMSE
##                       Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## Linear.Regression 6.789330 7.643416 8.054975 7.898578 8.297012 8.445941
## Elastic.Net       7.023522 7.389858 7.764023 7.728647 8.064425 8.326068
## PCR               7.189889 7.833668 7.996183 8.046503 8.231913 9.077769
## PLS               7.234305 7.790933 7.860970 7.925482 7.932825 8.673940
## MARS              7.378935 7.419016 7.682595 7.781426 8.121300 8.306568
##                   NA's
## Linear.Regression    0
## Elastic.Net          0
## PCR                  0
## PLS                  0
## MARS                 0
##
## Rsquared
##                        Min.   1st Qu.    Median      Mean   3rd Qu.
## Linear.Regression 0.2516288 0.2938432 0.3116055 0.3358196 0.3616757
## Elastic.Net       0.2783845 0.2958091 0.3472662 0.3593702 0.4137933
## PCR               0.1776466 0.2797380 0.2955079 0.3066535 0.3529264
## PLS               0.2514042 0.3040409 0.3231183 0.3259740 0.3569333
## MARS              0.2766183 0.3257554 0.3505151 0.3512856 0.3839220
##                        Max. NA's
## Linear.Regression 0.5056246    0
## Elastic.Net       0.4850429    0
## PCR               0.4442248    0
## PLS               0.4029653    0
## MARS              0.4191424    0
```

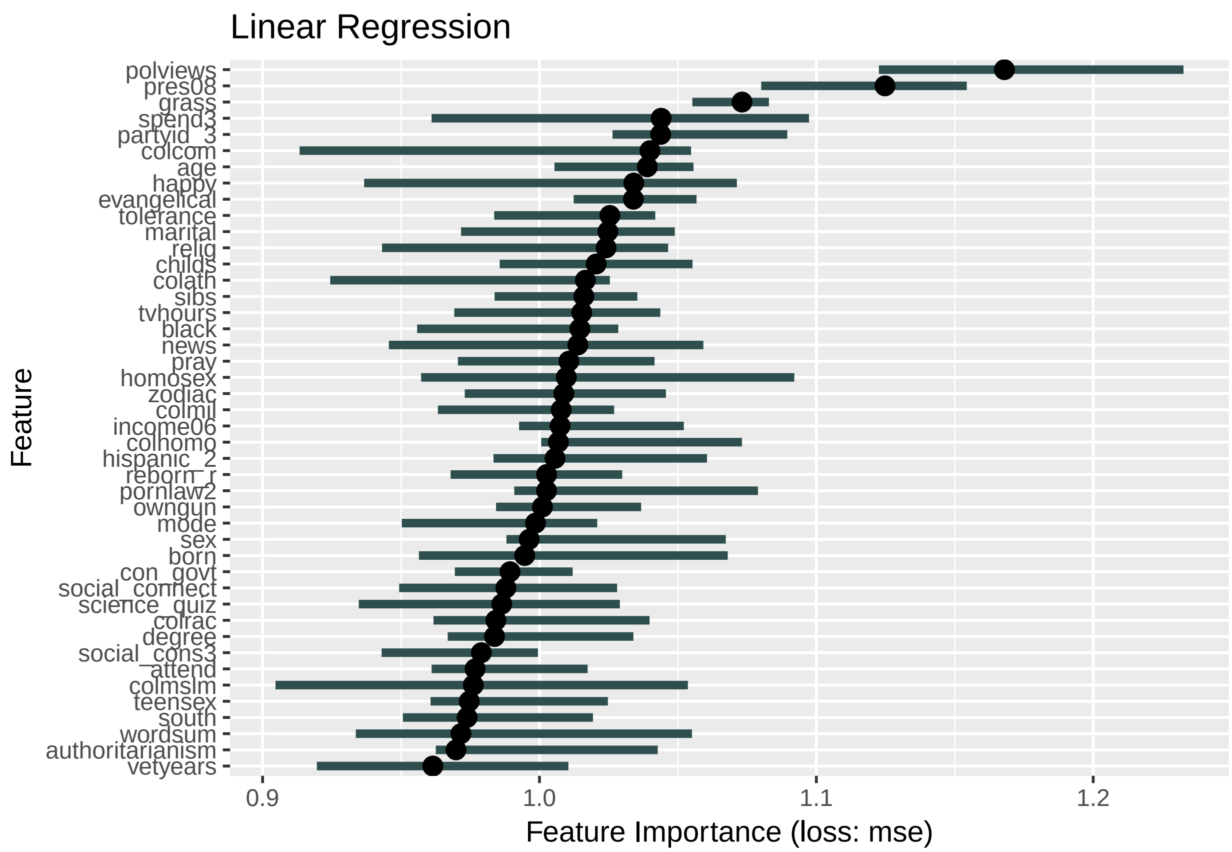
Looking at both RMSE and MAE, Elastic Net performed the best among all.

2. Apply model interpretation methods to each model. That is, for each model (the final tuned version), generate permutation-based feature importance plots, PDPs/ICE plots for the five most important variables, and feature interaction plots. Interpret the results with written analysis.

```
pred_lr = Predictor$new( model = lr, data = select(gss_train, -egalit_scale), y = gss_train$egalit_scale
pred_net = Predictor$new(model = ela.net,data = select(gss_train, -egalit_scale),y = gss_train$egalit_sc
pred_pcr = Predictor$new(model = pcr,data = select(gss_train, -egalit_scale),y = gss_train$egalit_scale
pred_pls = Predictor$new(model = pls,data = select(gss_train, -egalit_scale),y = gss_train$egalit_scale
pred_mars = Predictor$new(model = mars,data = select(gss_train, -egalit_scale),y = gss_train$egalit_scal
```

```
# Feature Importance
imp_lr = FeatureImp$new(pred_lr, loss = "mse")
imp_net = FeatureImp$new(pred_net, loss = "mse")
imp_pcr = FeatureImp$new(pred_pcr, loss = "mse")
imp_pls = FeatureImp$new(pred_pls, loss = "mse")
imp_mars = FeatureImp$new(pred_mars, loss = "mse")

img1 = plot(imp_lr) + ggtitle("Linear Regression")
img2 = plot(imp_pcr) + ggtitle("PCR")
img3 = plot(imp_pls) + ggtitle("PLS")
img4 = plot(imp_net) + ggtitle("Elastic net")
img5 = plot(imp_mars) + ggtitle("MARS")
```

```
img1
```

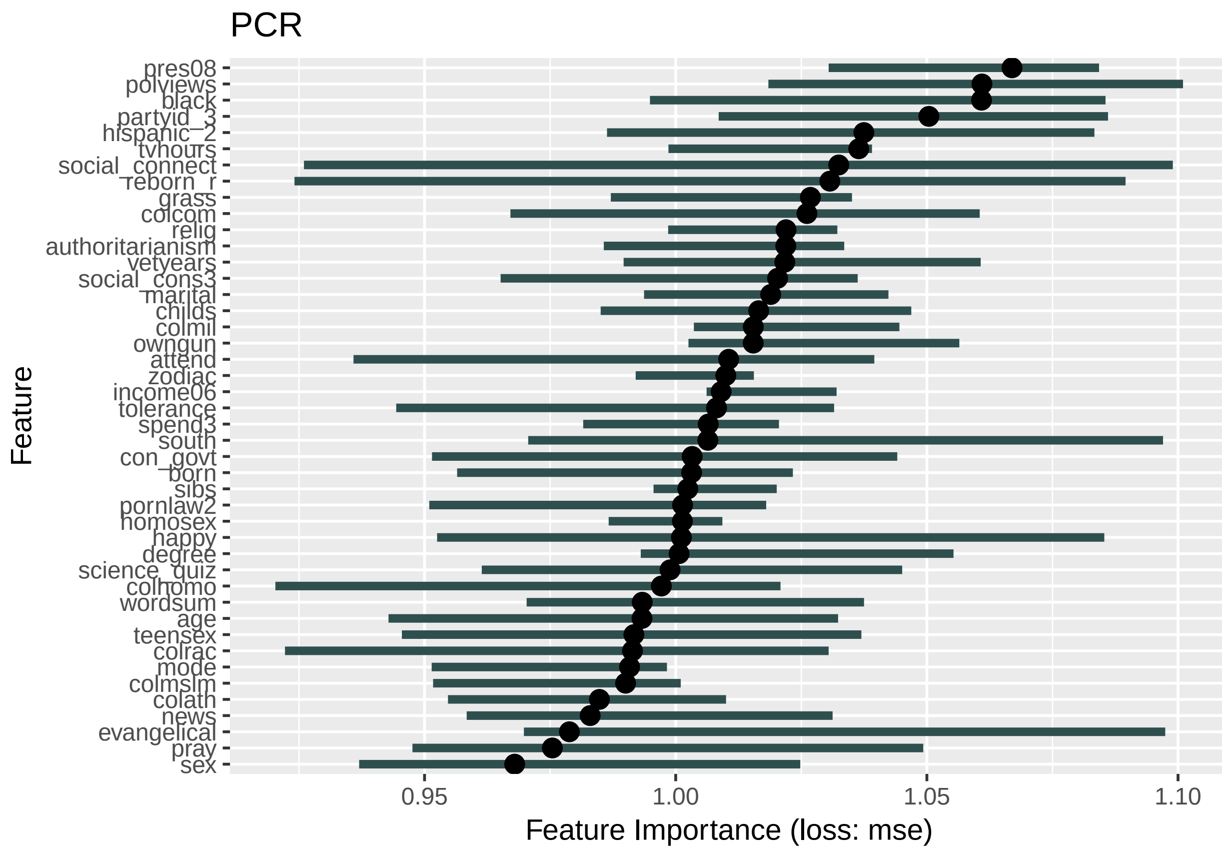



```
head(imp_lr$results, 5)
```

```
##      feature importance.05 importance importance.95 permutation.error
```

```
## 1  polviews     1.1225923   1.167918      1.232604          63.56455
## 2    pres08     1.0800850   1.124796      1.154340          61.21763
## 3     grass     1.0552063   1.073156      1.082864          58.40708
## 4    spend3     0.9610456   1.043940      1.097356          56.81702
## 5 partyid_3     1.0263772   1.043772      1.089503          56.80784
```

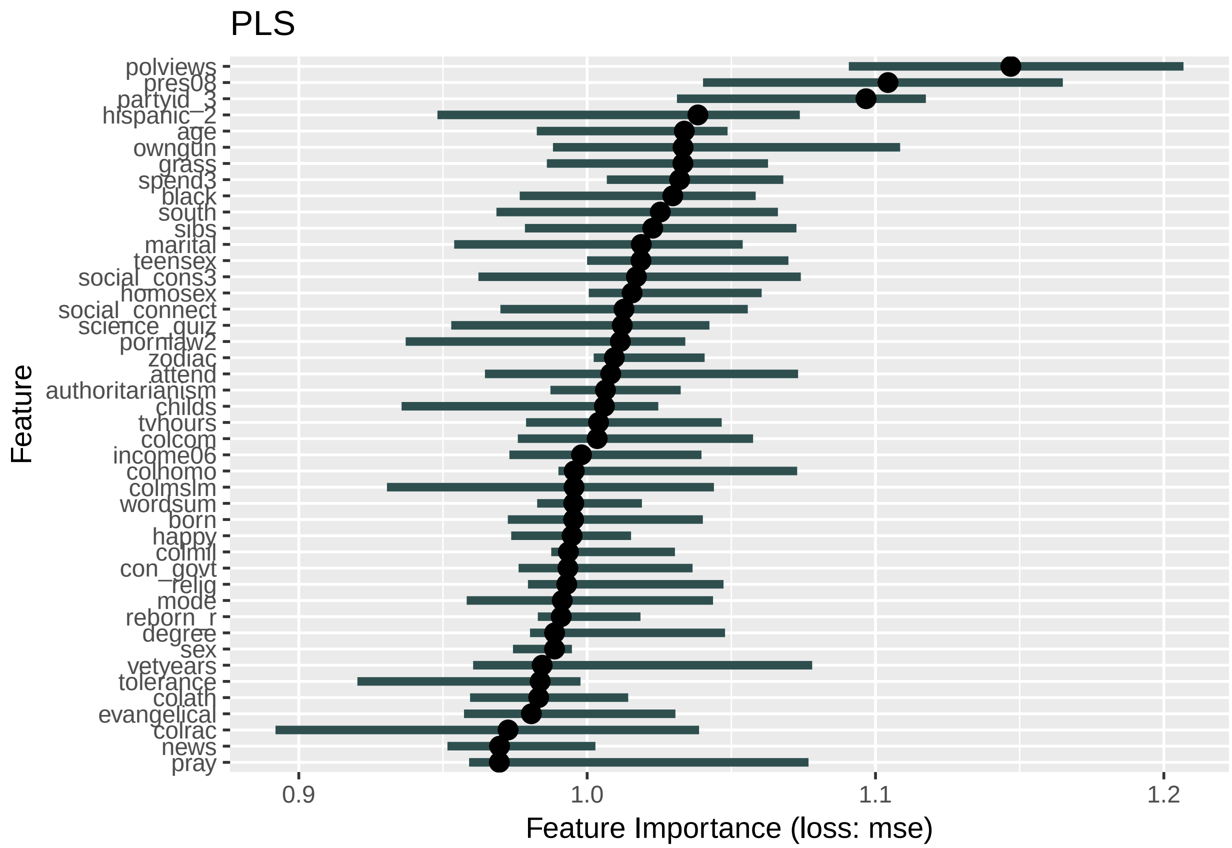
```r
img2
```




```r
head(imp_net$results, 5)
```

```
##     feature importance.05 importance importance.95 permutation.error
## 1    pres08     1.1220595   1.178630      1.205459          67.18219
## 2  polviews     1.0599580   1.134426      1.170556          64.66255
## 3 partyid_3     1.0183967   1.063340      1.087106          60.61060
## 4    degree     0.9799325   1.053507      1.060483          60.05011
## 5      news     0.9377634   1.045942      1.053519          59.61895
```

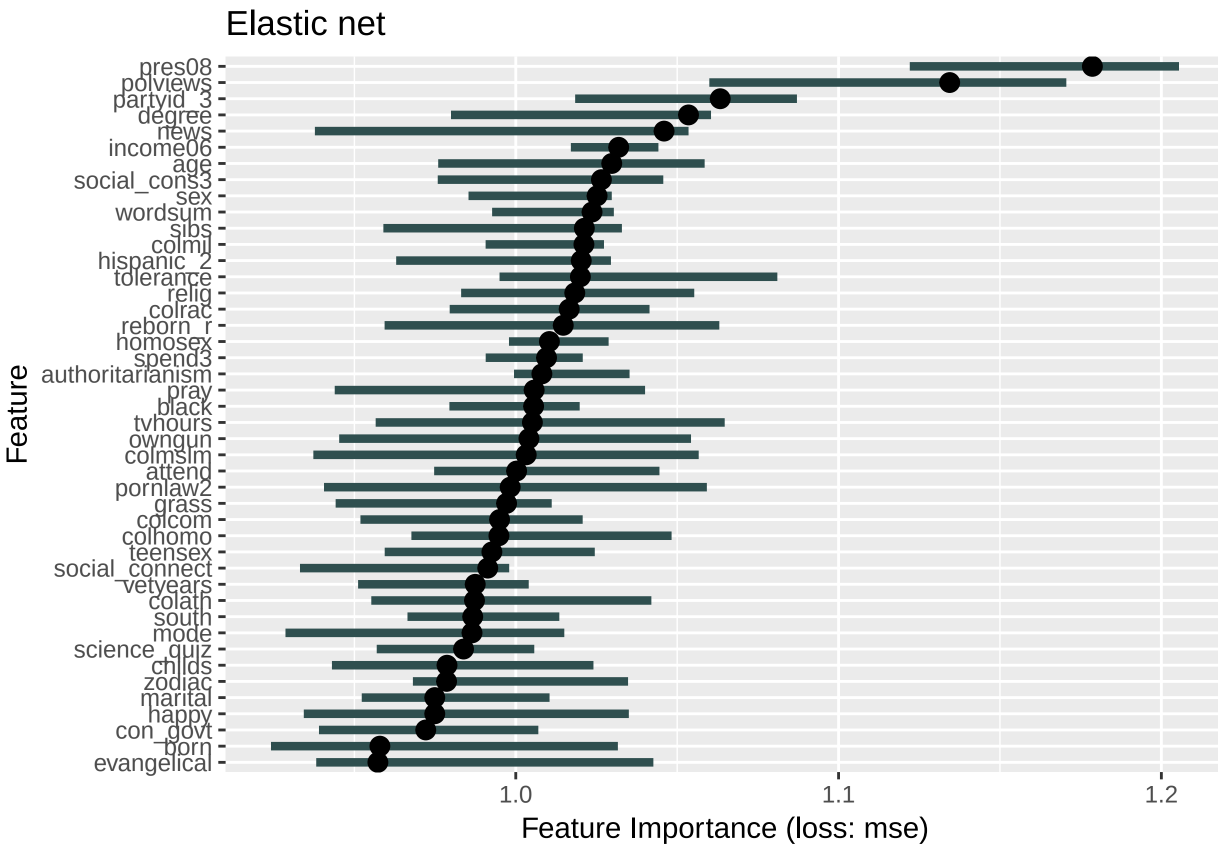
```r
img3
```




```r
head(imp_pcr$results, 5)
```

```
##       feature importance.05 importance importance.95 permutation.error
## 1      pres08     1.0304483   1.066949      1.084274          67.35683
## 2    polviews     1.0184507   1.060965      1.100990          66.97907
## 3       black     0.9948723   1.060885      1.085568          66.97399
## 4  partyid_3     1.0085498   1.050395      1.086067          66.31179
## 5 hispanic_2     0.9863257   1.037460      1.083355          65.49519
```

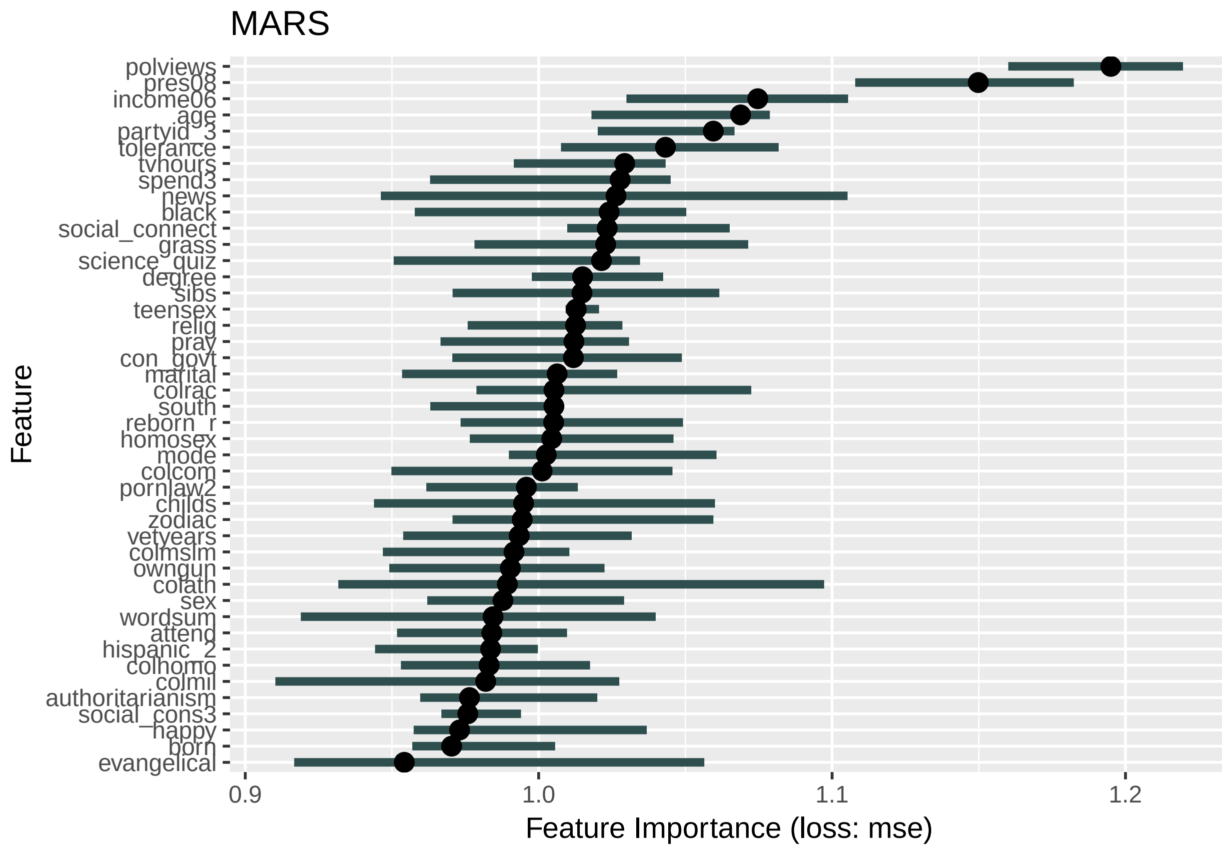
```r
img4
```




```
head(imp_pls$results, 5)
```

```
##      feature importance.05 importance importance.95 permutation.error
## 1   polviews     1.0907496   1.146941      1.206812          65.07771
## 2     pres08     1.0401969   1.104331      1.164965          62.65998
## 3  partyid_3     1.0311443   1.096719      1.117473          62.22808
## 4 hispanic_2     0.9480884   1.038417      1.073760          58.92003
## 5        age     0.9825345   1.033695      1.048706          58.65210
```

```
img5
```




```r
head(imp_mars$results, 5)
```

```
##     feature importance.05 importance importance.95 permutation.error
## 1  polviews      1.160049   1.195020      1.219601          66.74976
## 2    pres08      1.107883   1.149817      1.182385          64.22484
## 3  income06      1.029918   1.074666      1.105460          60.02719
## 4       age      1.017997   1.068831      1.078793          59.70123
## 5 partyid_3      1.020129   1.059531      1.066752          59.18180
```

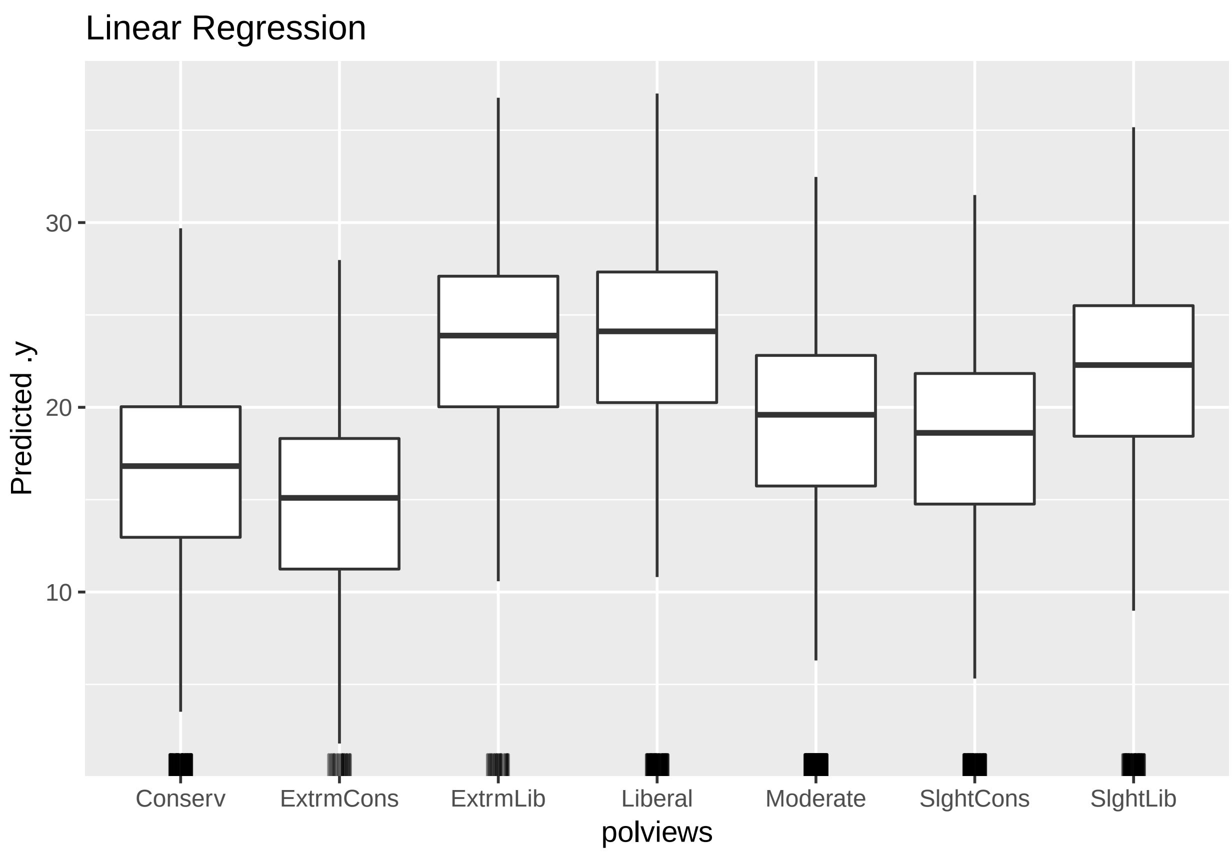
In general, we can find that polviews, pres08 are the most important two features for all of these five model settings; other important features include: partyid_3, age and income06. I will draw PDP on these variables

```r
preds = tibble(name = c("Linear Regression", "PCR", "PLS", "Elastic Net", "MARS"),
  models = list(Linear.Regression = pred_lr,
                Elastic.net = pred_net,
                PCR = pred_pcr,
                PLS = pred_pls,
                MARS = pred_mars
))

predictors_pdp <- preds %>%
mutate(
  polviews = map2(models, name, ~ FeatureEffect$new(.x, "polviews", method = "pdp+ice") %>%
  plot() + ggtitle(.y)),
  pres08 = map2(models, name, ~ FeatureEffect$new(.x, "pres08", method = "pdp+ice") %>%
  plot() + ggtitle(.y)),
  partyid_3 = map2(models, name, ~ FeatureEffect$new(.x, "partyid_3",method = "pdp+ice") %>%
```

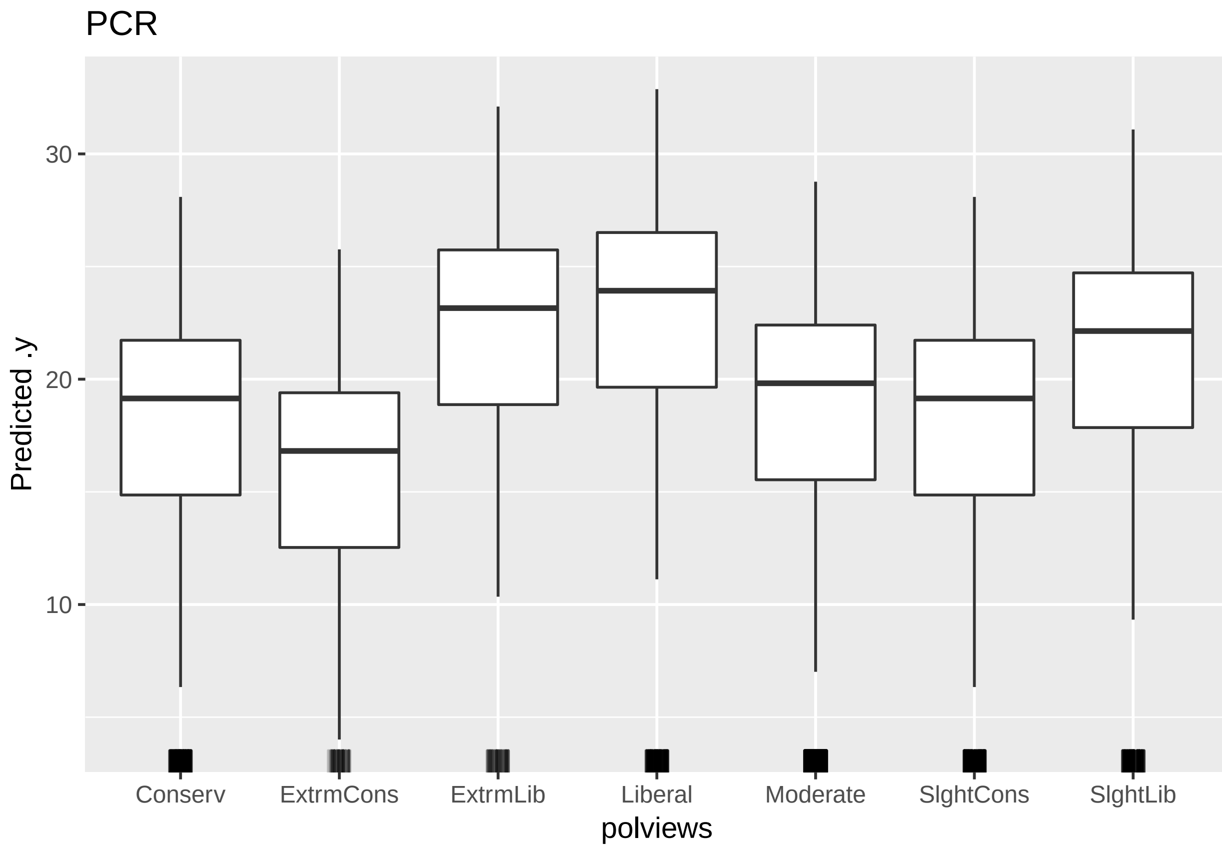
```
  plot() + ggtitle(.y)),
  age = map2(models, name, ~ FeatureEffect$new(.x, "age", method = "pdp+ice", center.at = min(gss_train$
  plot() + ggtitle(.y)),
  inc06 = map2(models, name, ~ FeatureEffect$new(.x, "income06", method = "pdp+ice", center.at = min(gss
  plot() + ggtitle(.y))
)
```

```
predictors_pdp$polviews
```

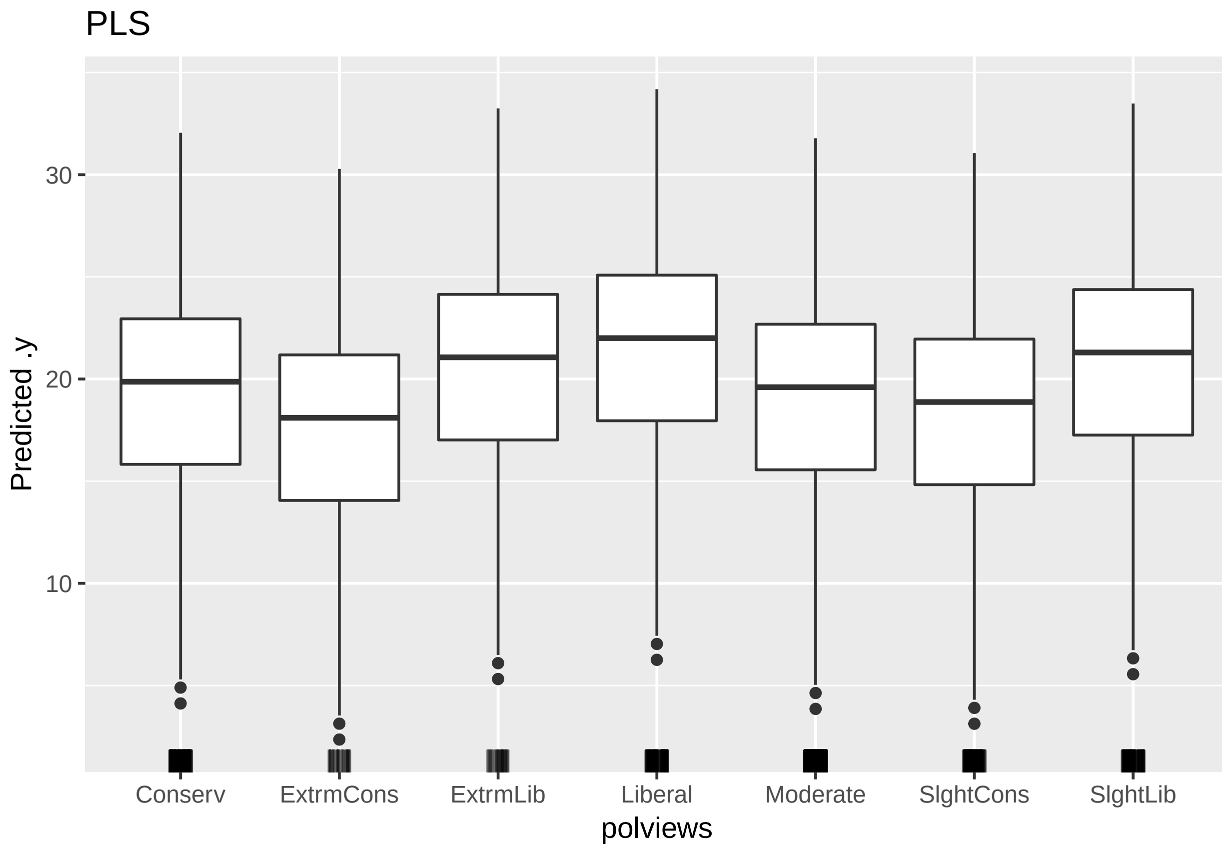
```
## $Linear.Regression
```



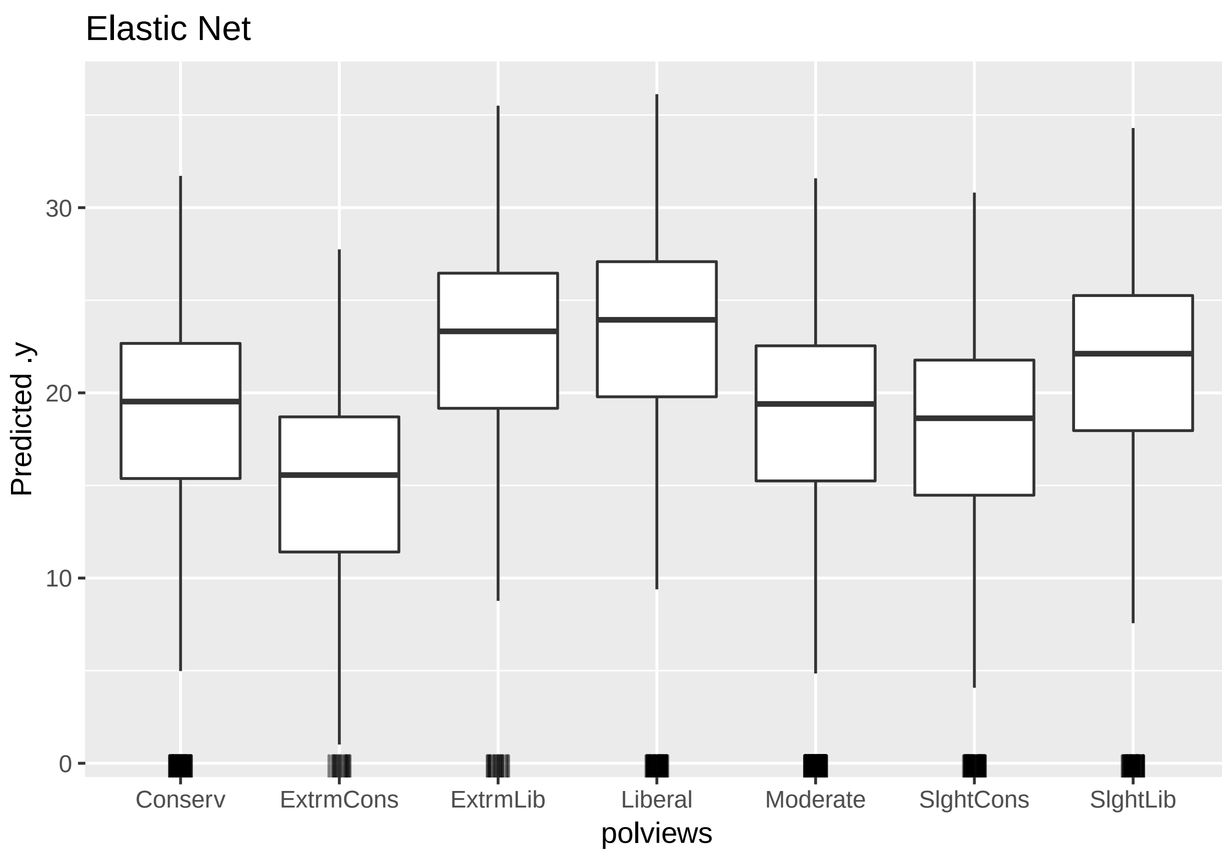

```
##
## $Elastic.net
```


PCR
Predicted .y
30
20
10
Conserv
ExtrmCons
ExtrmLib
Liberal
Moderate
SlghtCons
SlghtLib
polviews

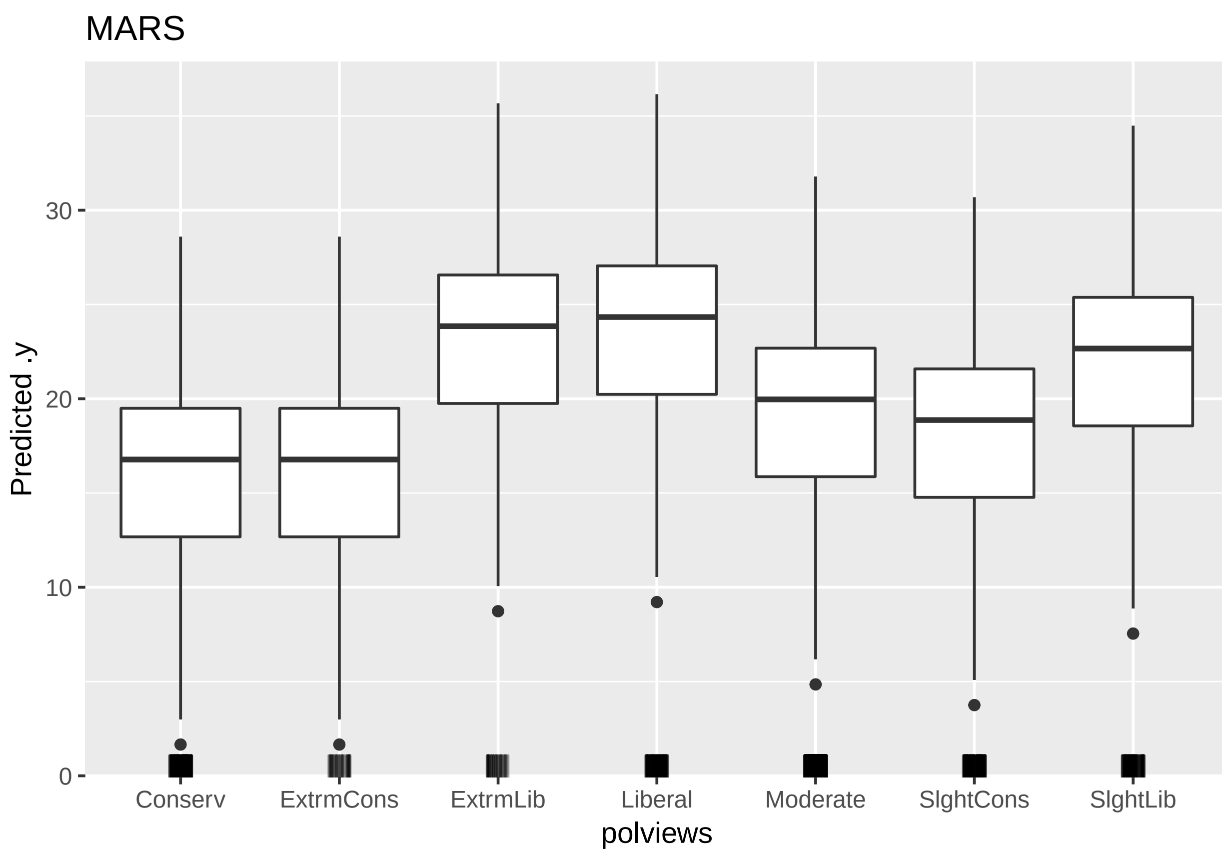
```
##
## $PCR
```


PLS
Predicted .y
30
20
10
Conserv
ExtrmCons
ExtrmLib
Liberal
Moderate
SlghtCons
SlghtLib
polviews

```
##
## $PLS
```

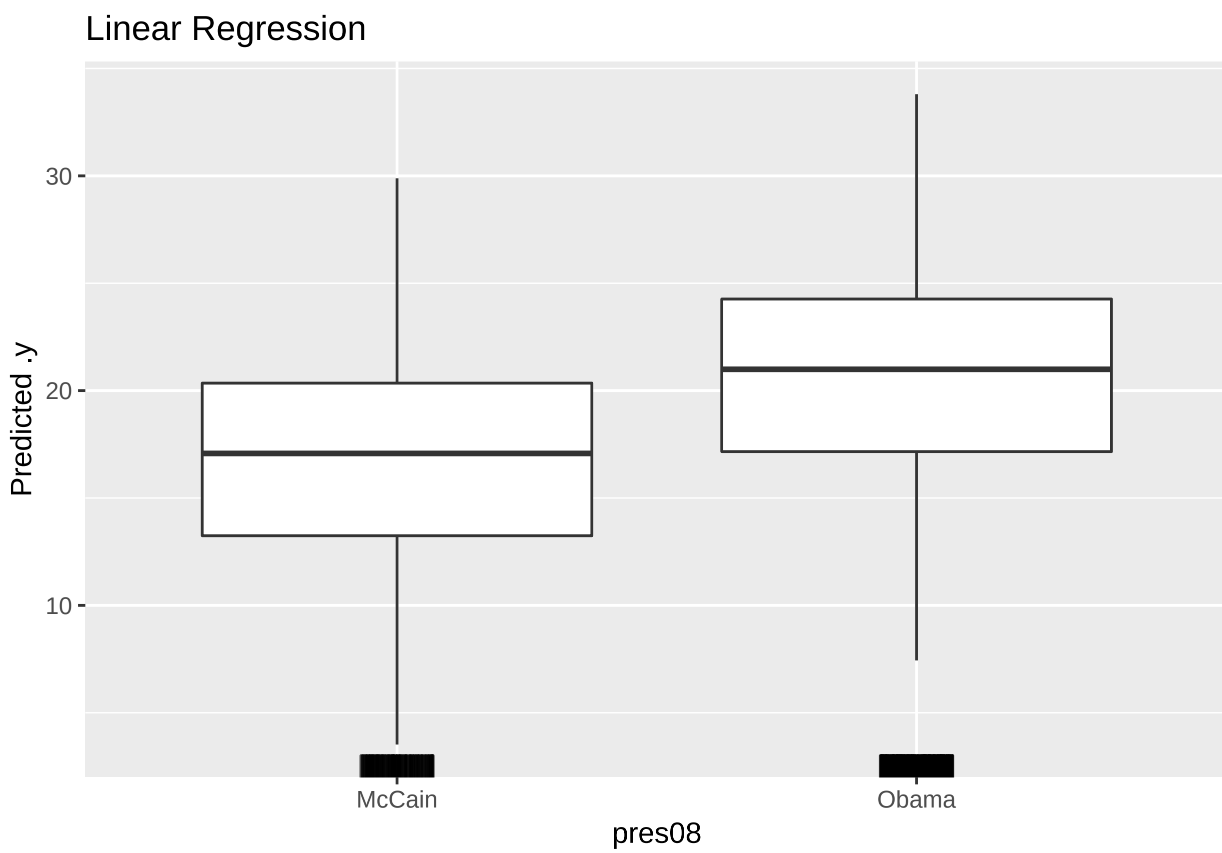

Elastic Net
Predicted .y
polviews
30
20
10
0
Conserv
ExtrmCons
ExtrmLib
Liberal
Moderate
SlghtCons
SlghtLib

```
##
## $MARS
```

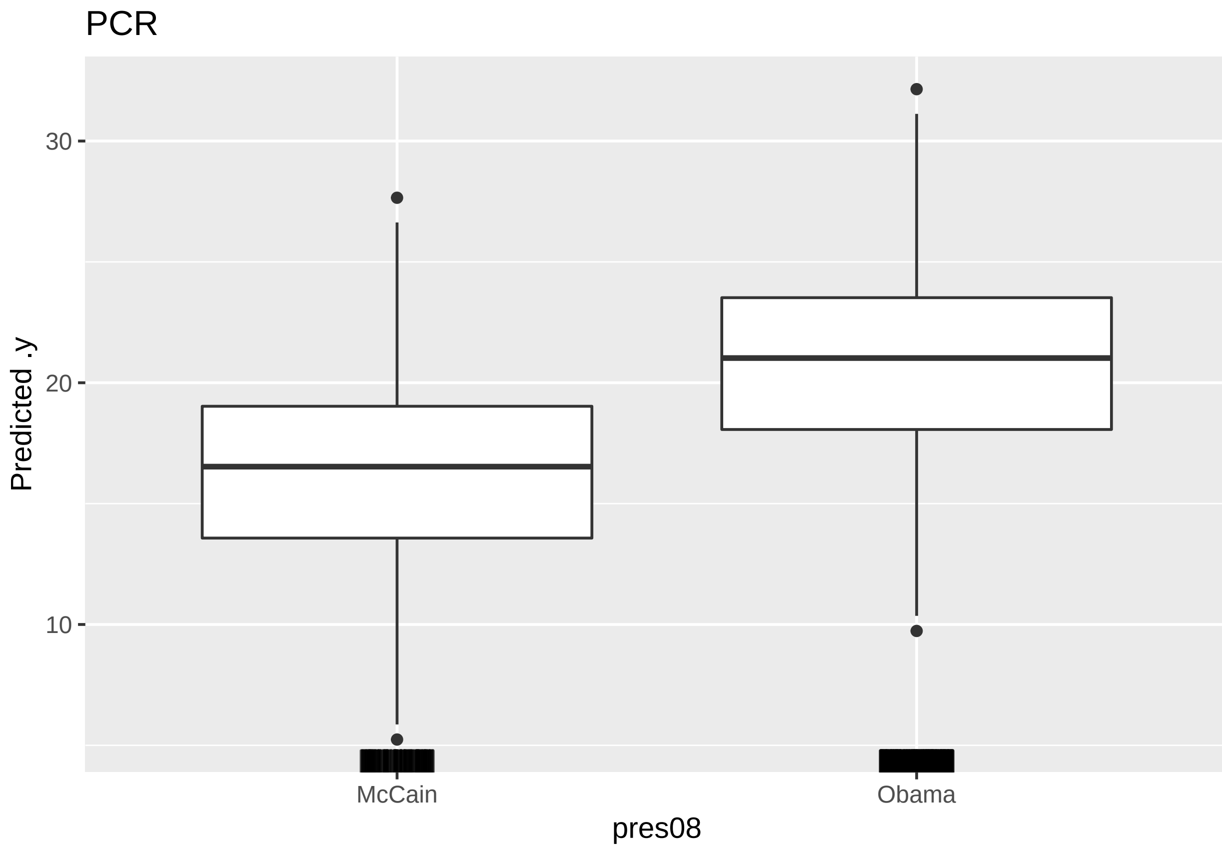



```
predictors_pdp$pres08
```

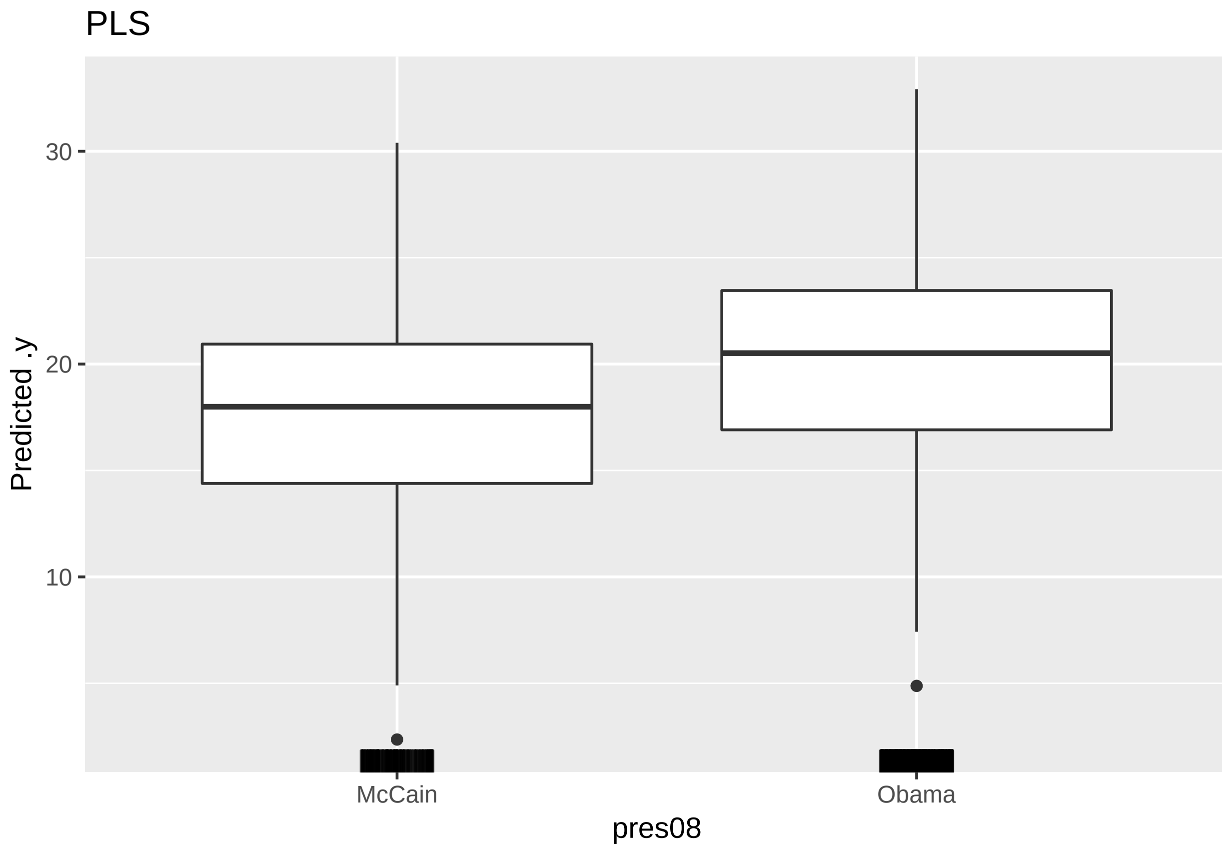
```
## $Linear.Regression
```



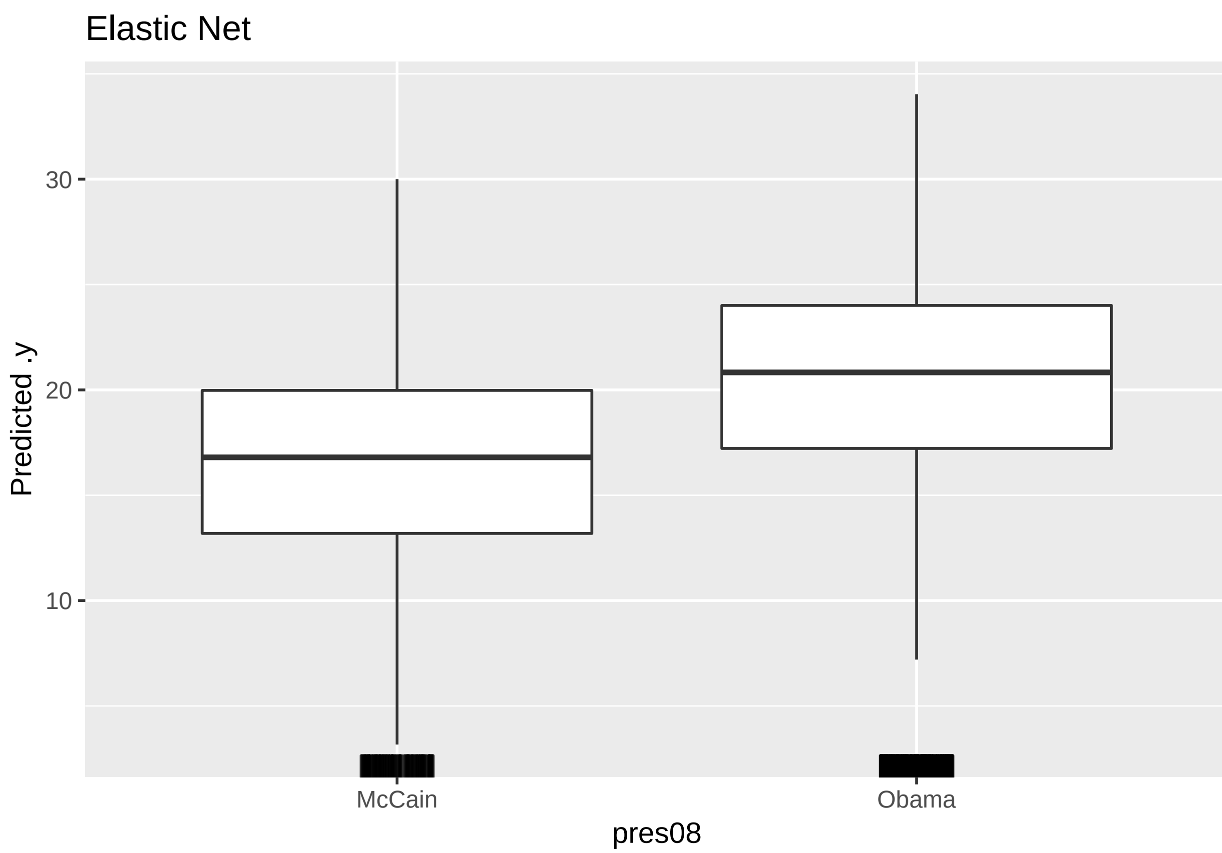

```
##
## $Elastic.net
```



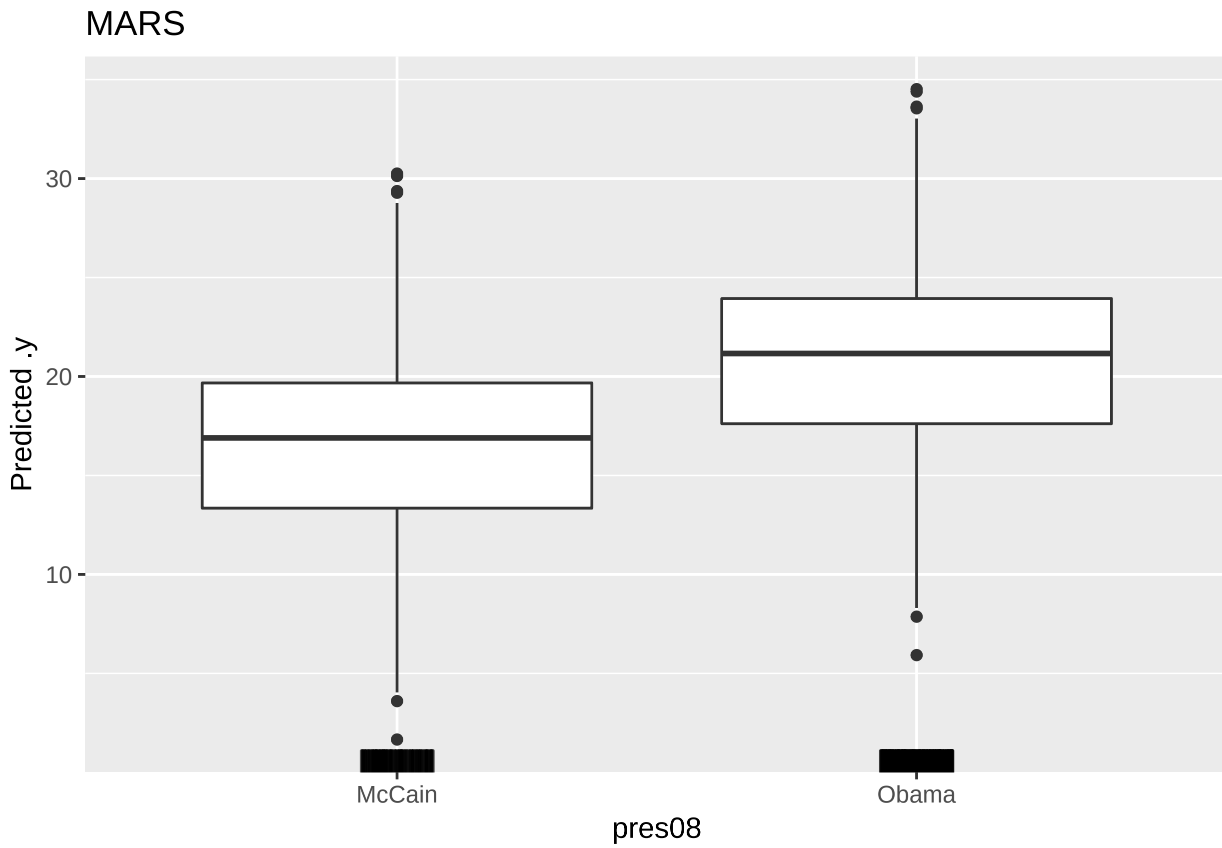

```
##
## $PCR
```




```
##
## $PLS
```

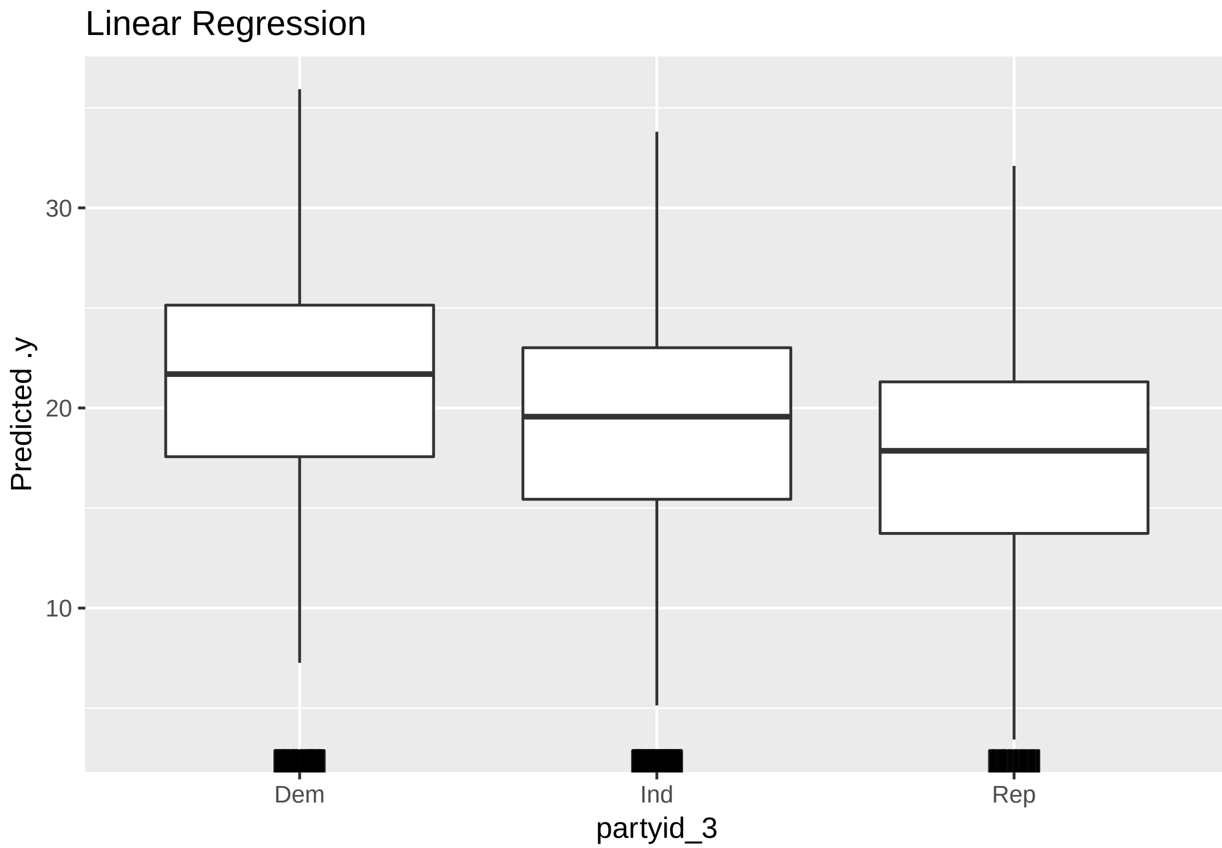



```
##
## $MARS
```

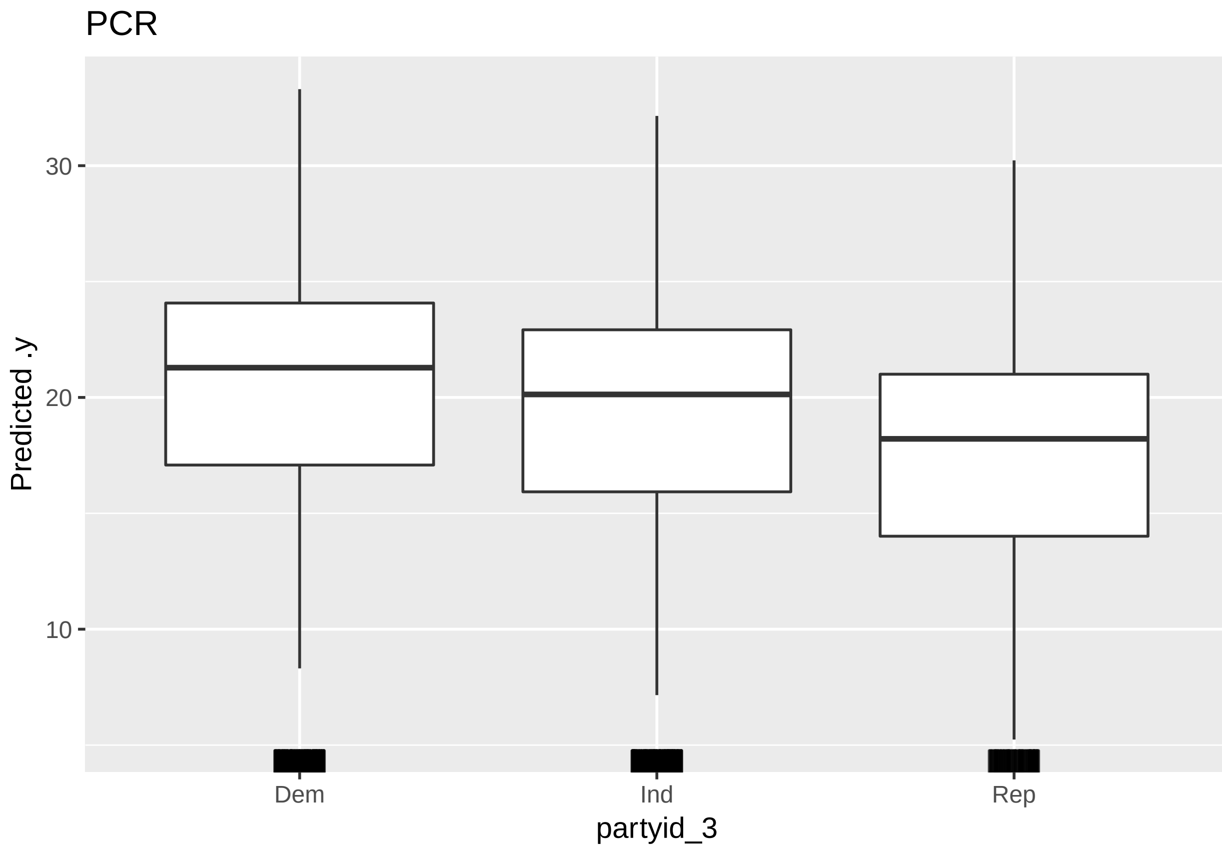



```
predictors_pdp$partyid_3
```

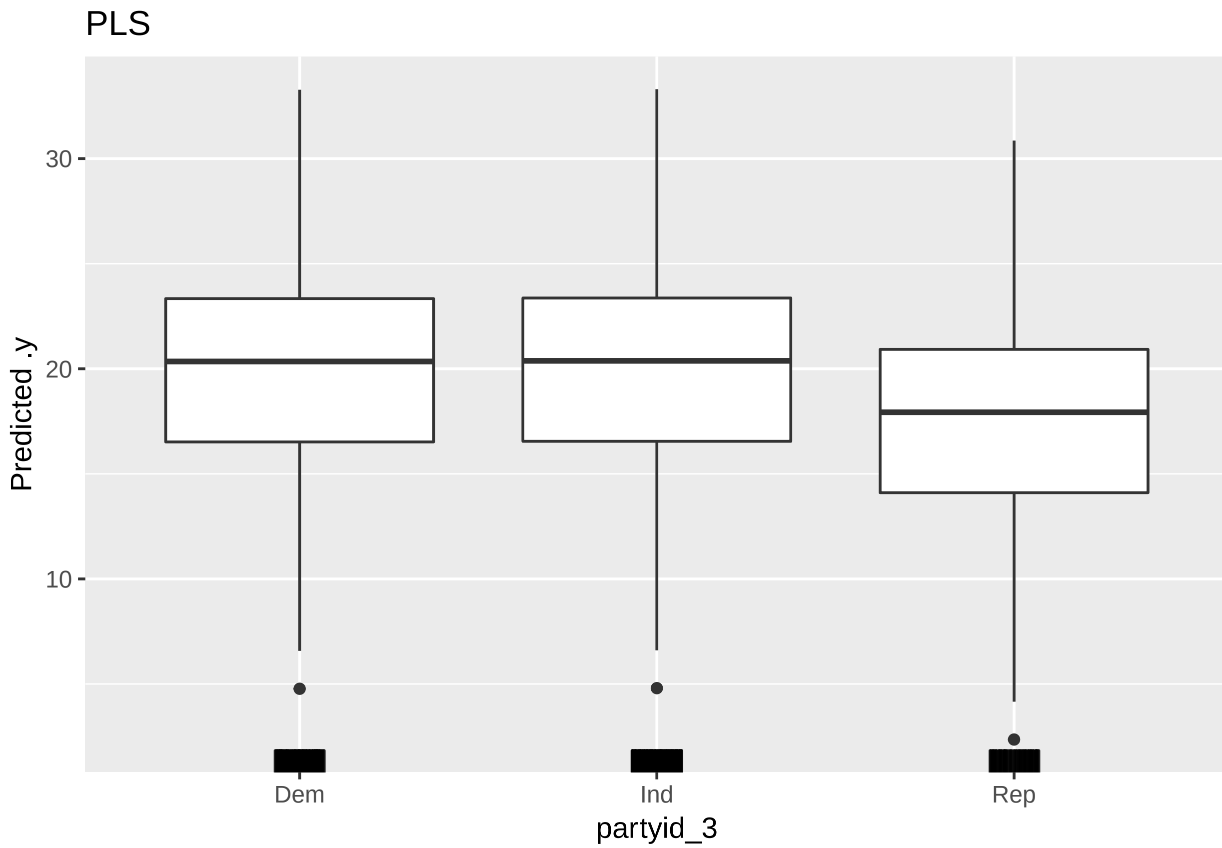
```
## $Linear.Regression
```



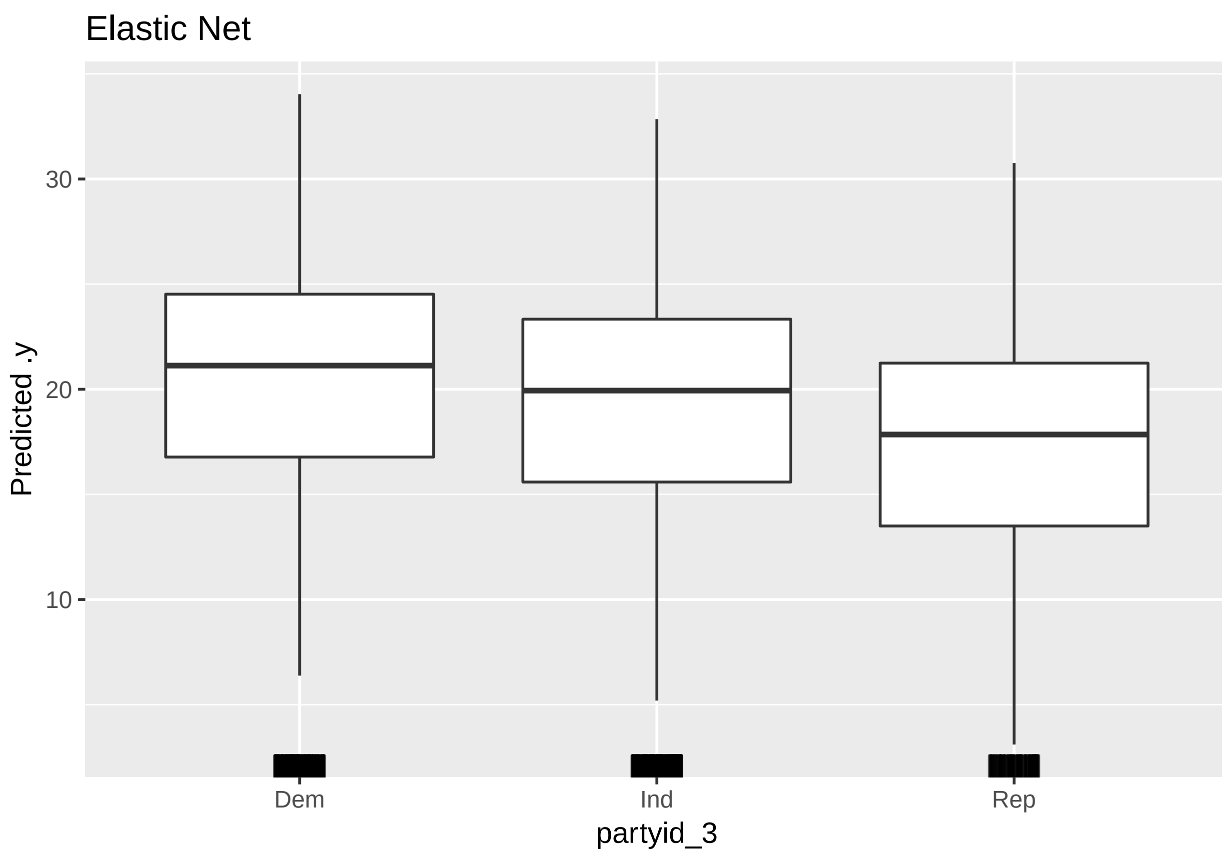

```
##
## $Elastic.net
```



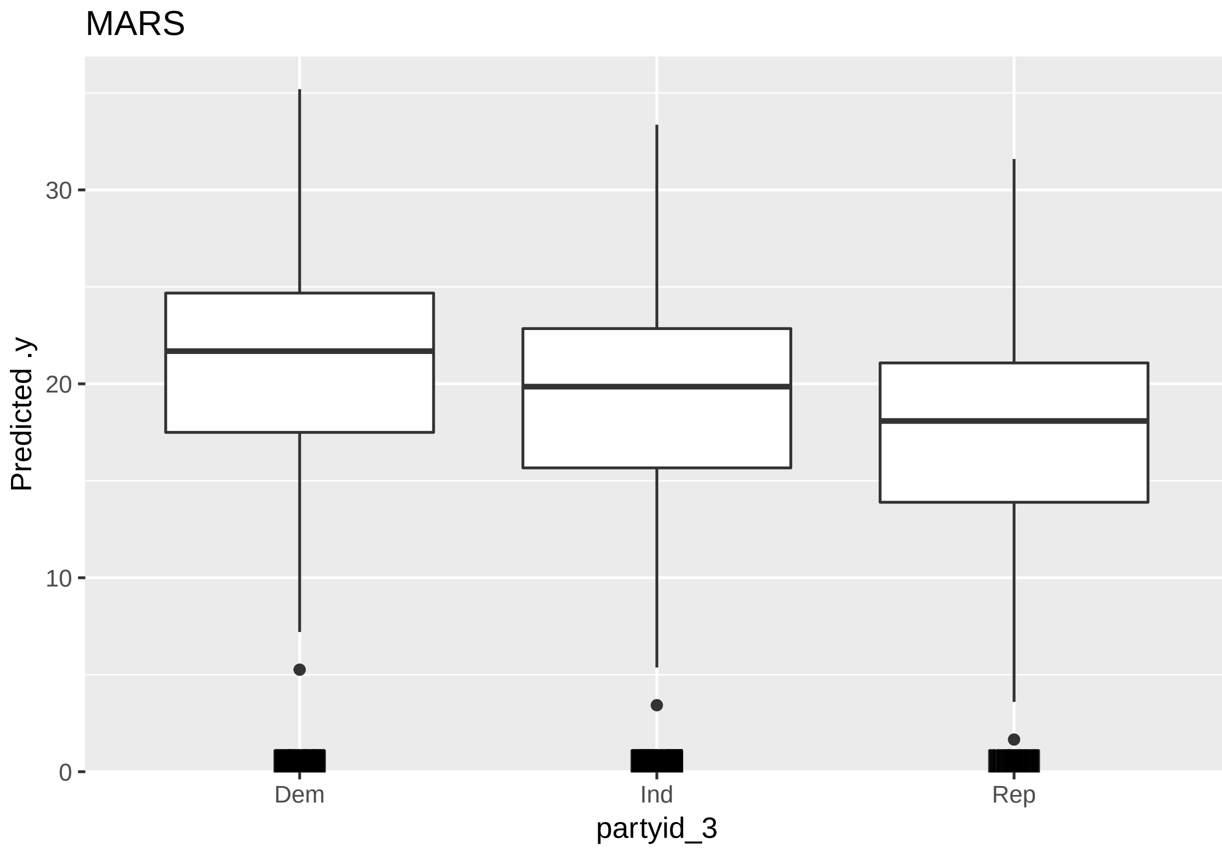

```
##
## $PCR
```




```
##
## $PLS
```



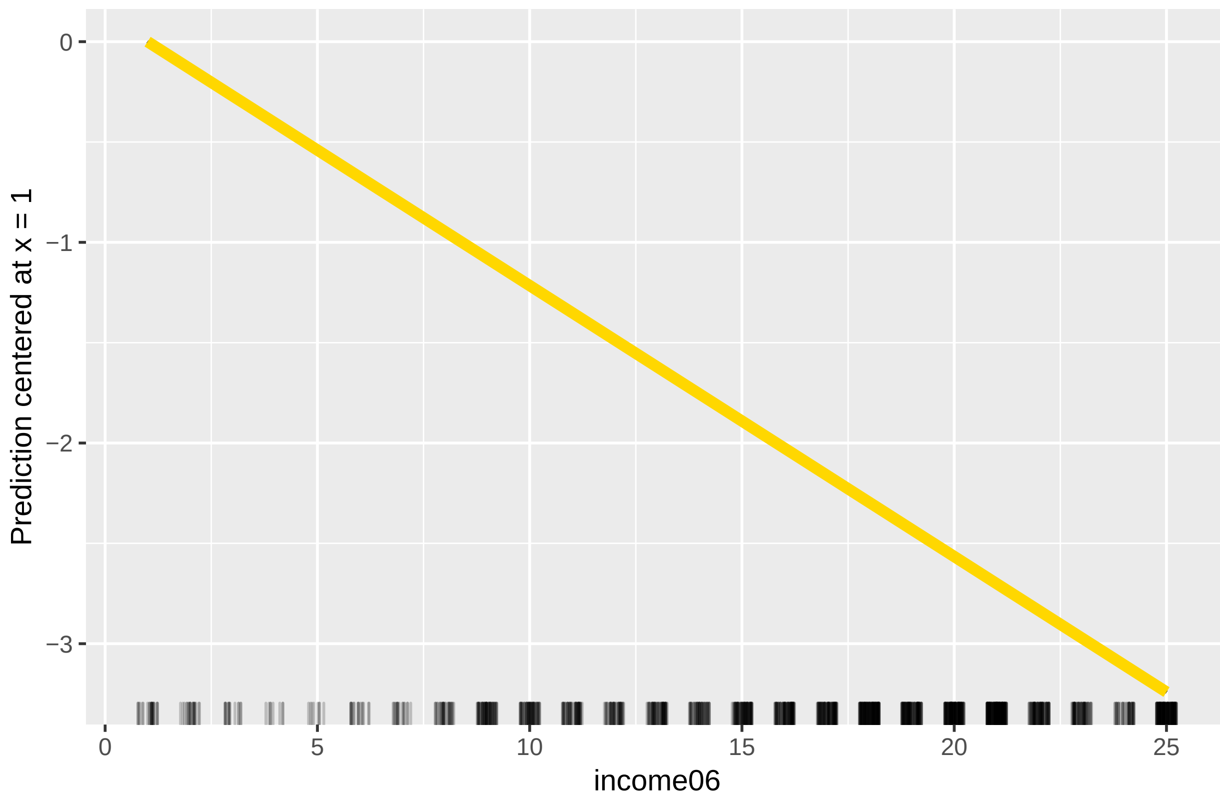

```
##
## $MARS
```



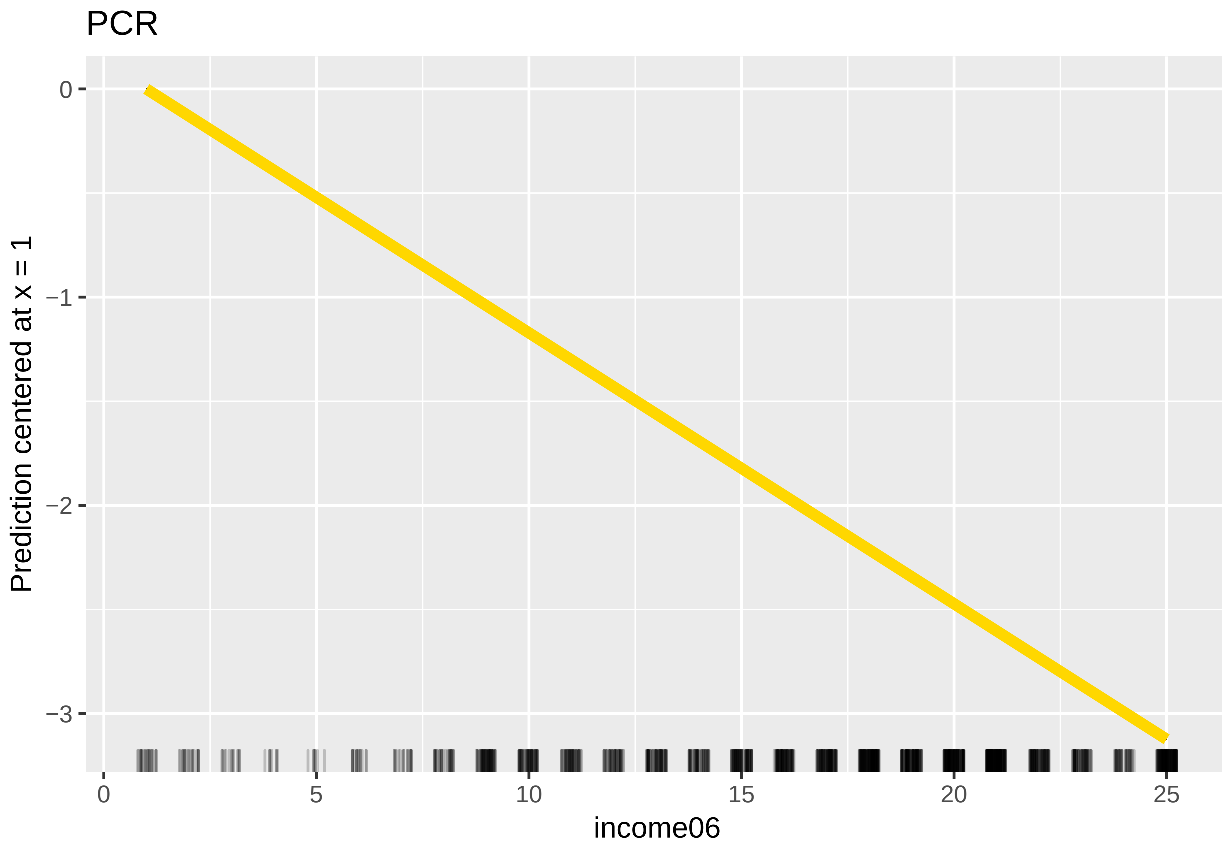

```
predictors_pdp$inc06
```

```
## $Linear.Regression
```

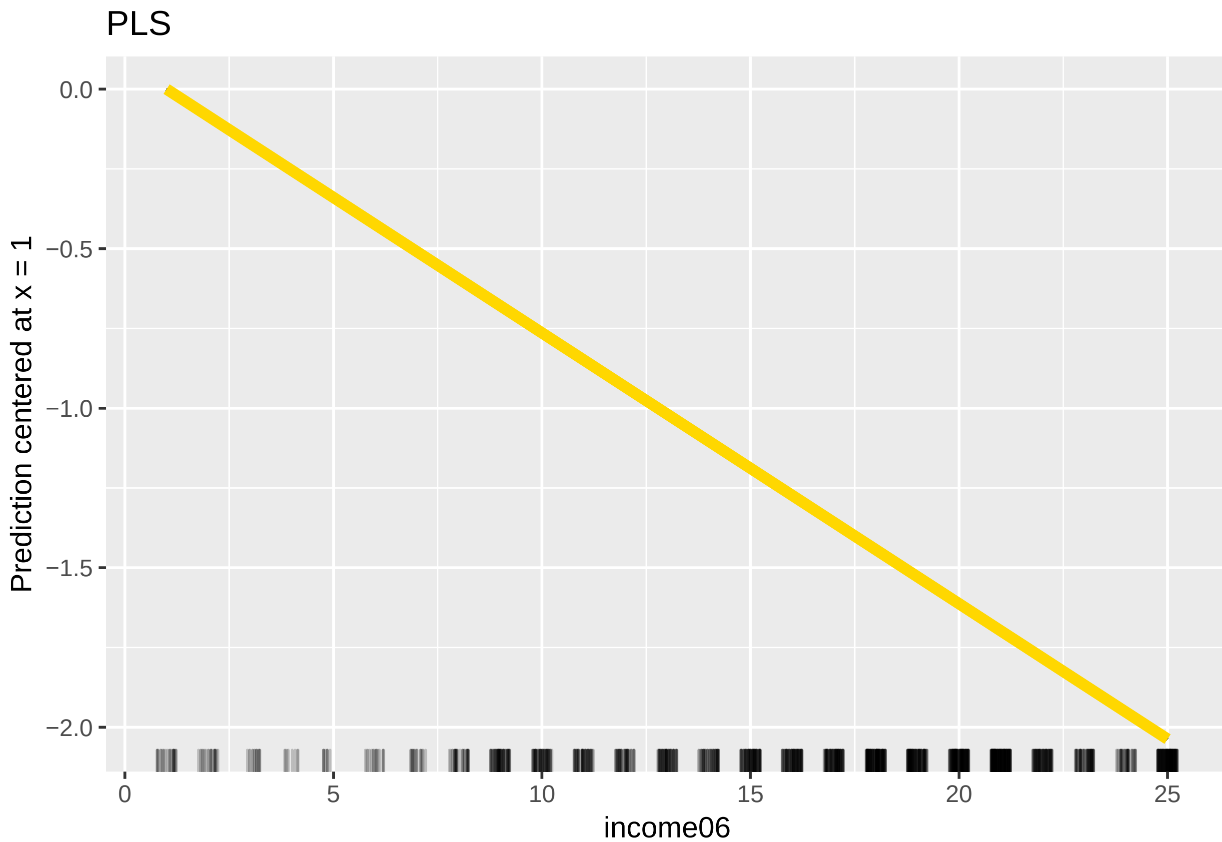
Linear Regression



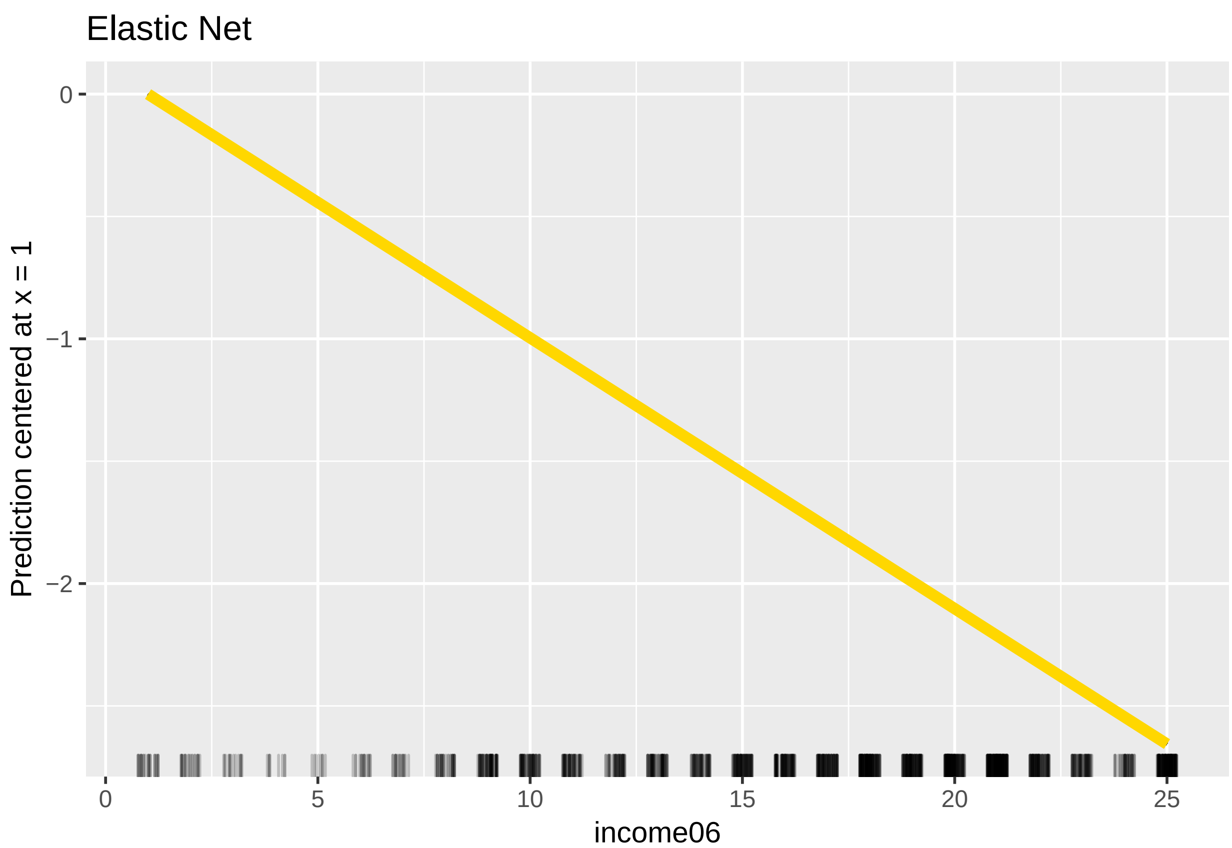

```
##
## $Elastic.net
```




```
##
## $PCR
```


PLS
0.0
−0.5
−1.0
−1.5
−2.0
Prediction centered at x = 1
0
5
10
15
20
25
income06

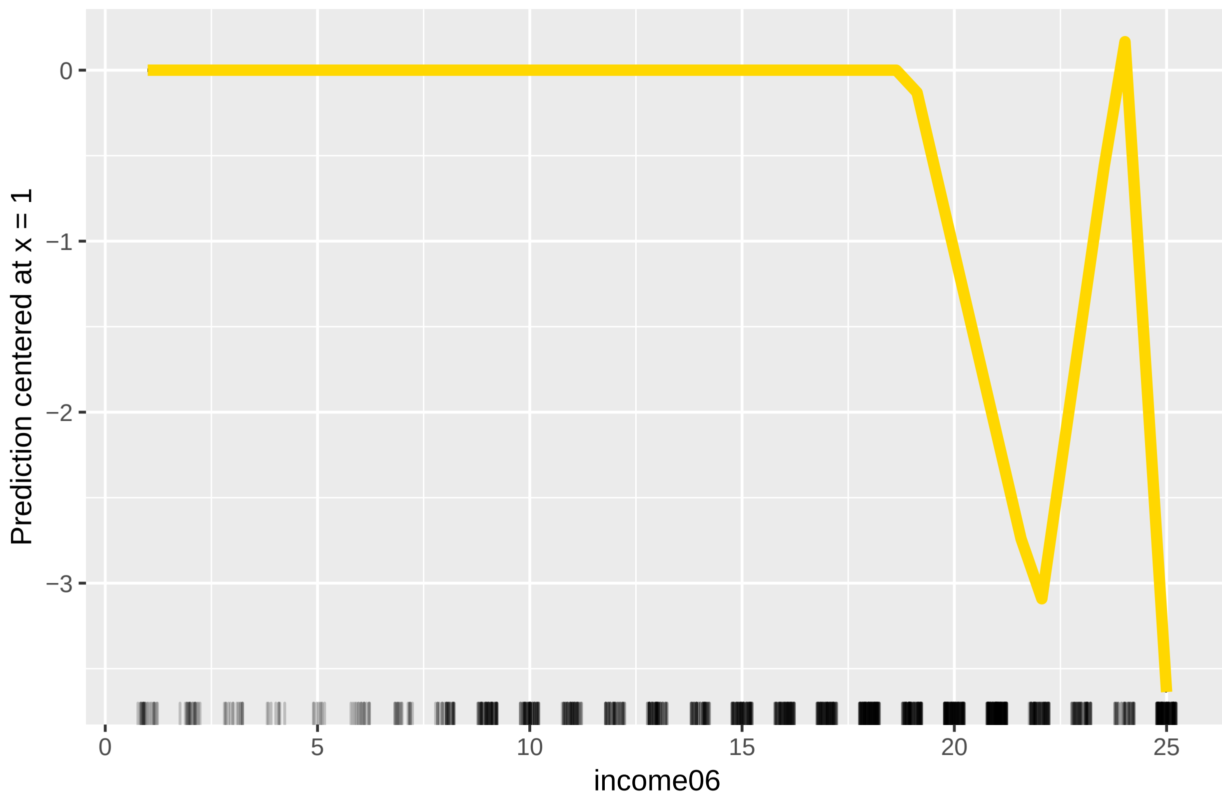
```
##
## $PLS
```



Elastic Net
0
−1
−2
Prediction centered at x = 1
0
5
10
15
20
25
income06

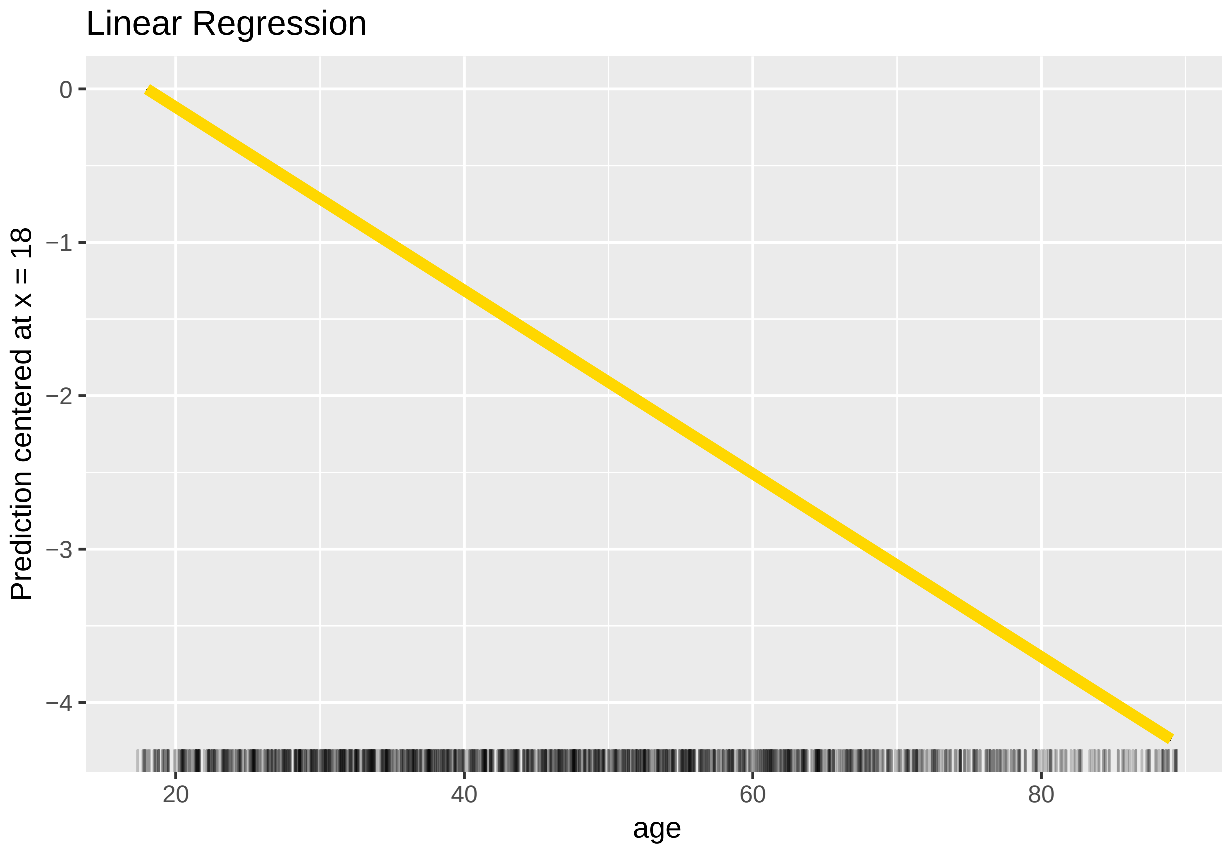

```
##
## $MARS
```

MARS



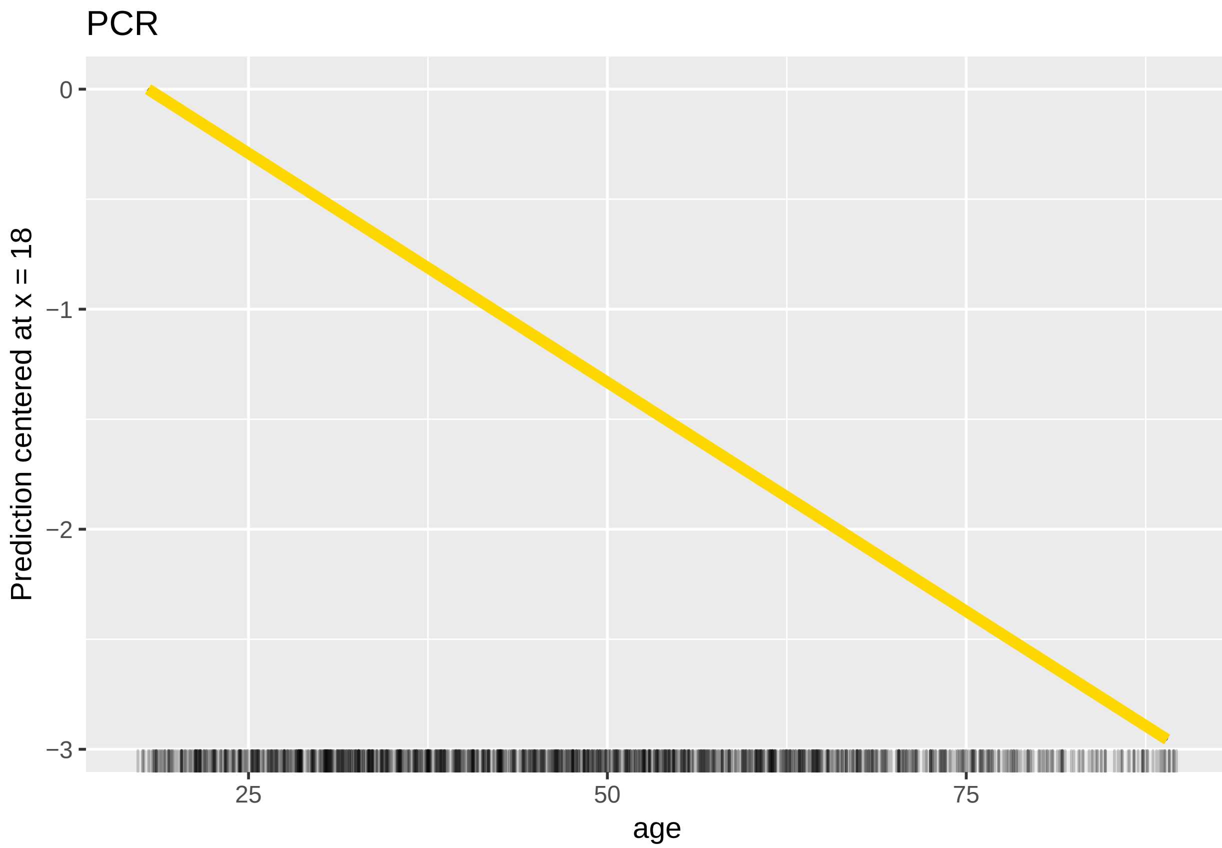

```
predictors_pdp$age
```

```
## $Linear.Regression
```

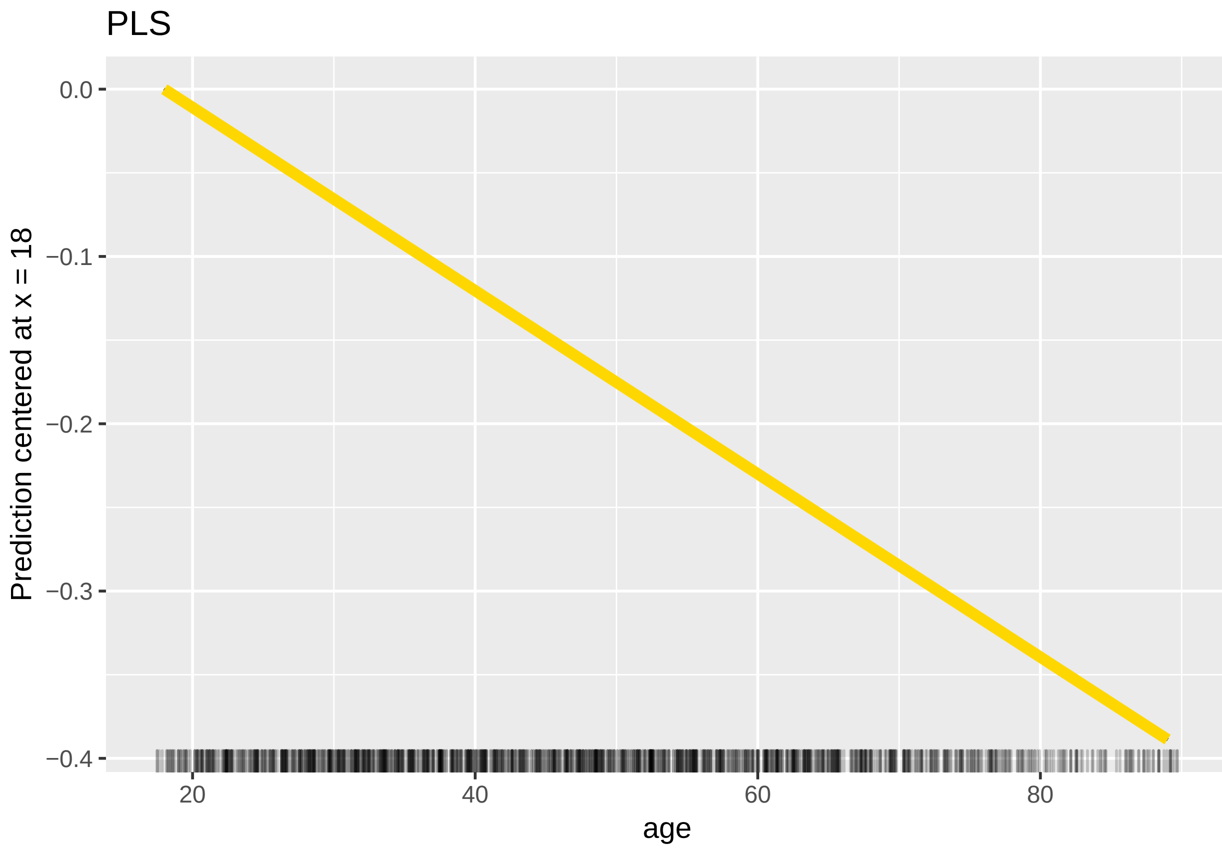


Linear Regression
Prediction centered at x = 18
0
−1
−2
−3
−4
20
40
60
80
age

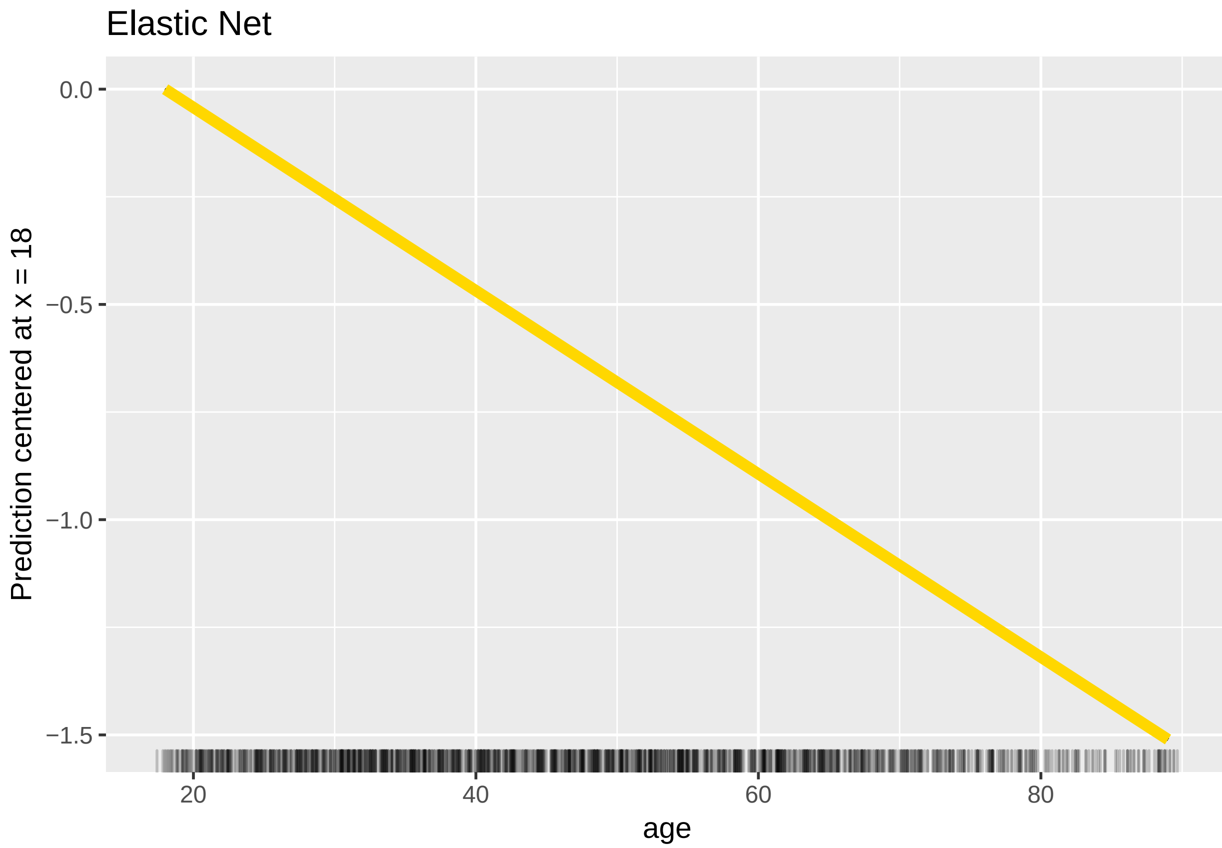

```
##
## $Elastic.net
```



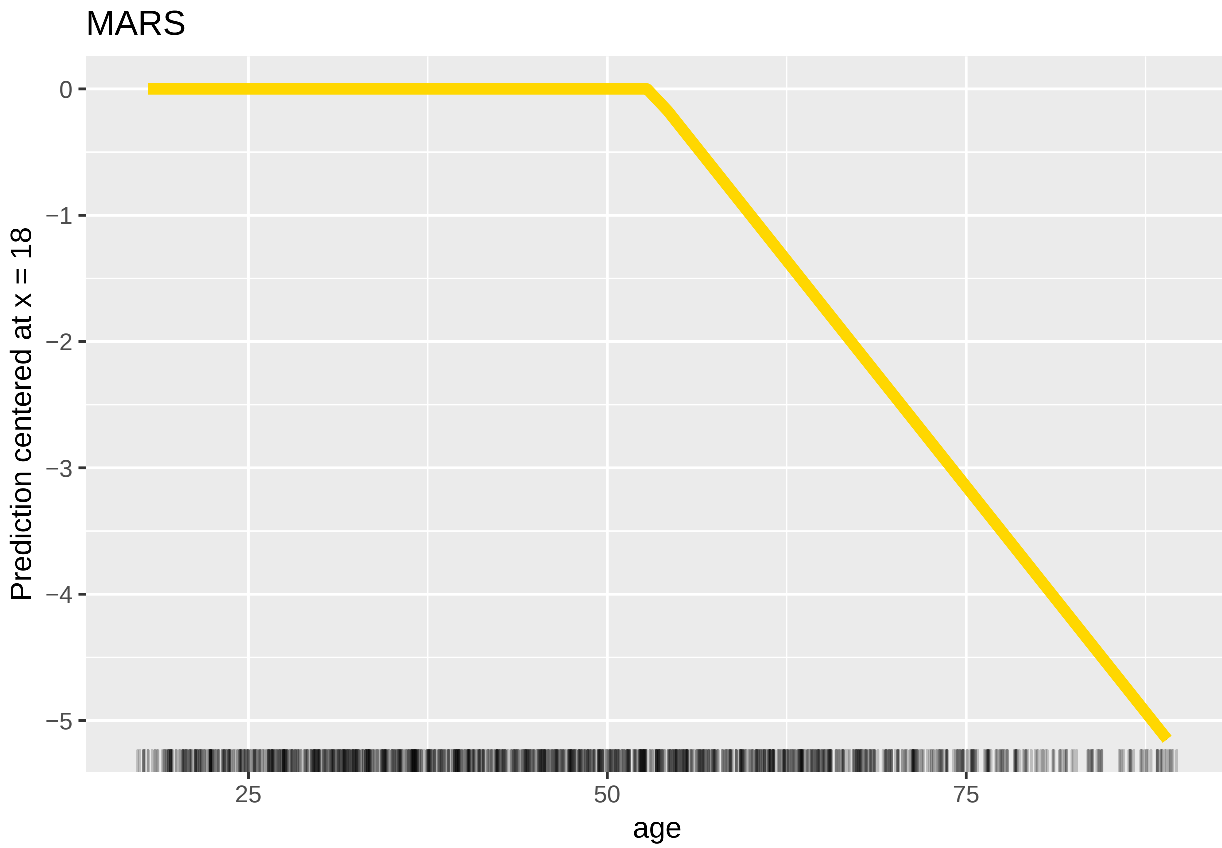

```
##
## $PCR
```




```
##
## $PLS
```

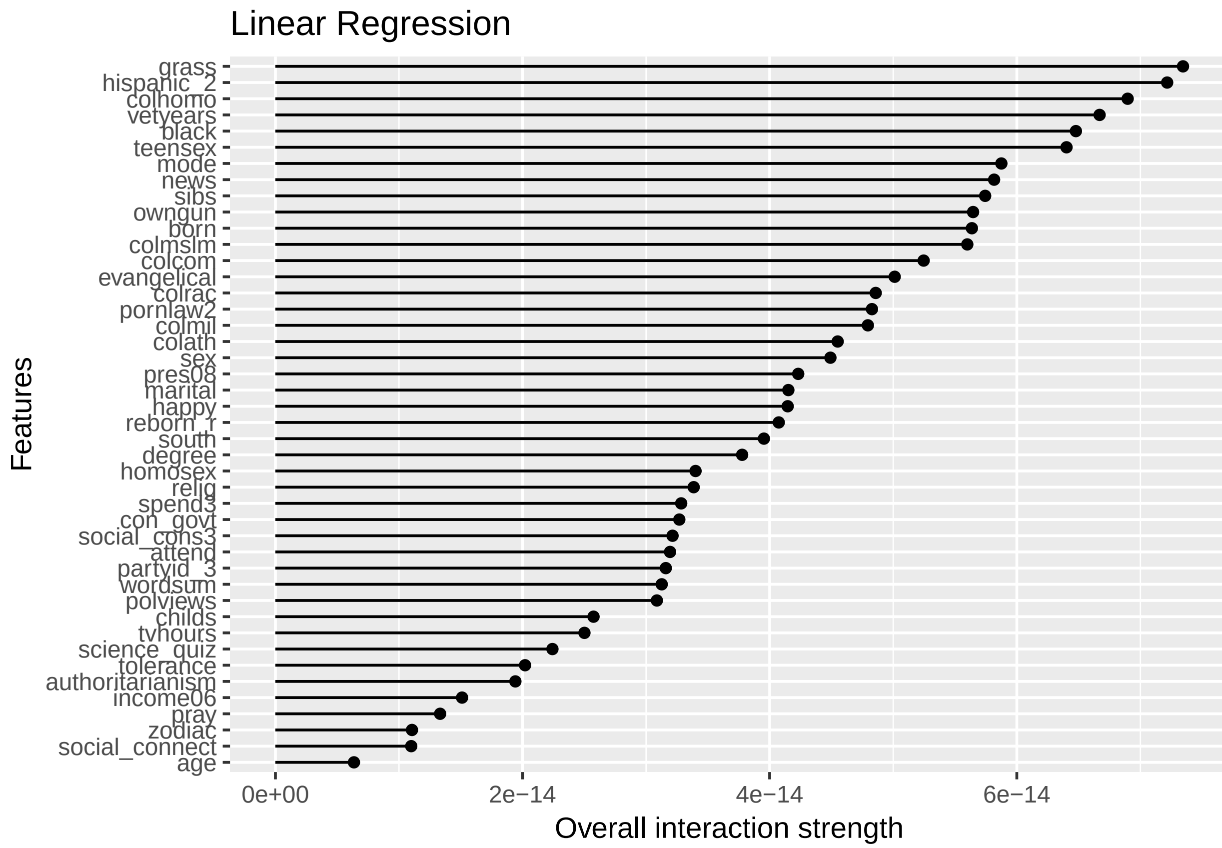

Elastic Net
0.0
−0.5
−1.0
−1.5
Prediction centered at x = 18
20
40
60
80
age

```
##
## $MARS
```



From these PDPs, we can find that: * In general, the more liberal the interviewees are, the more egalitarianism they hold. * Those who voted for Obama are more prone to be ealitarian. * Democrats favor more egalitarianism. * In general, the more income people earn, the less egalitarian they are. * In general, the older people get, the less egalitarian they are.

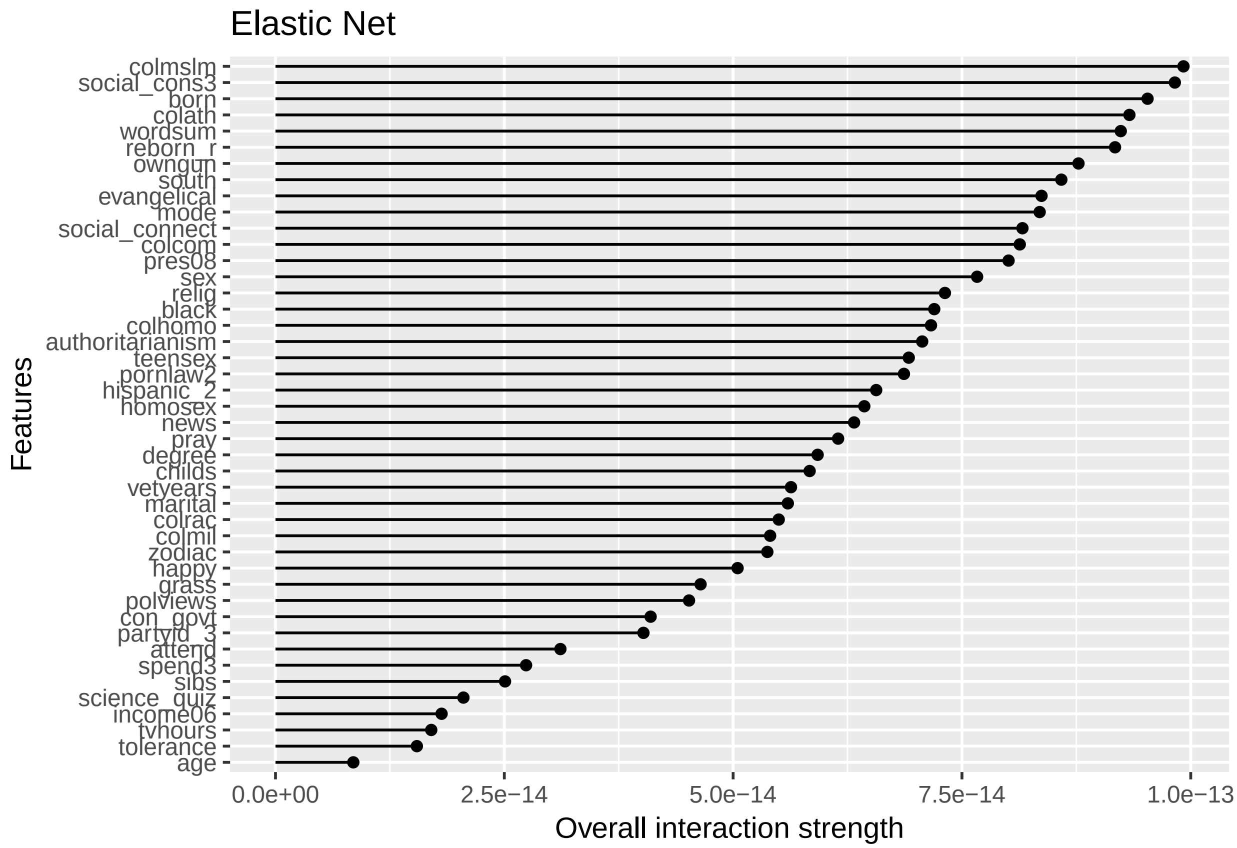
```
# Linear Regression feature interaction
lr_int = Interaction$new(pred_lr)
lr_int_score = lr_int$results
plot(lr_int) + ggtitle("Linear Regression")
```




```r
lr_int_score %>% arrange(-.interaction) %>% head(5)
```

```
##      .feature .interaction
## 1       grass 7.344880e-14
## 2 hispanic_2 7.216728e-14
## 3     colhomo 6.897090e-14
## 4    vetyears 6.669839e-14
## 5       black 6.478122e-14
```

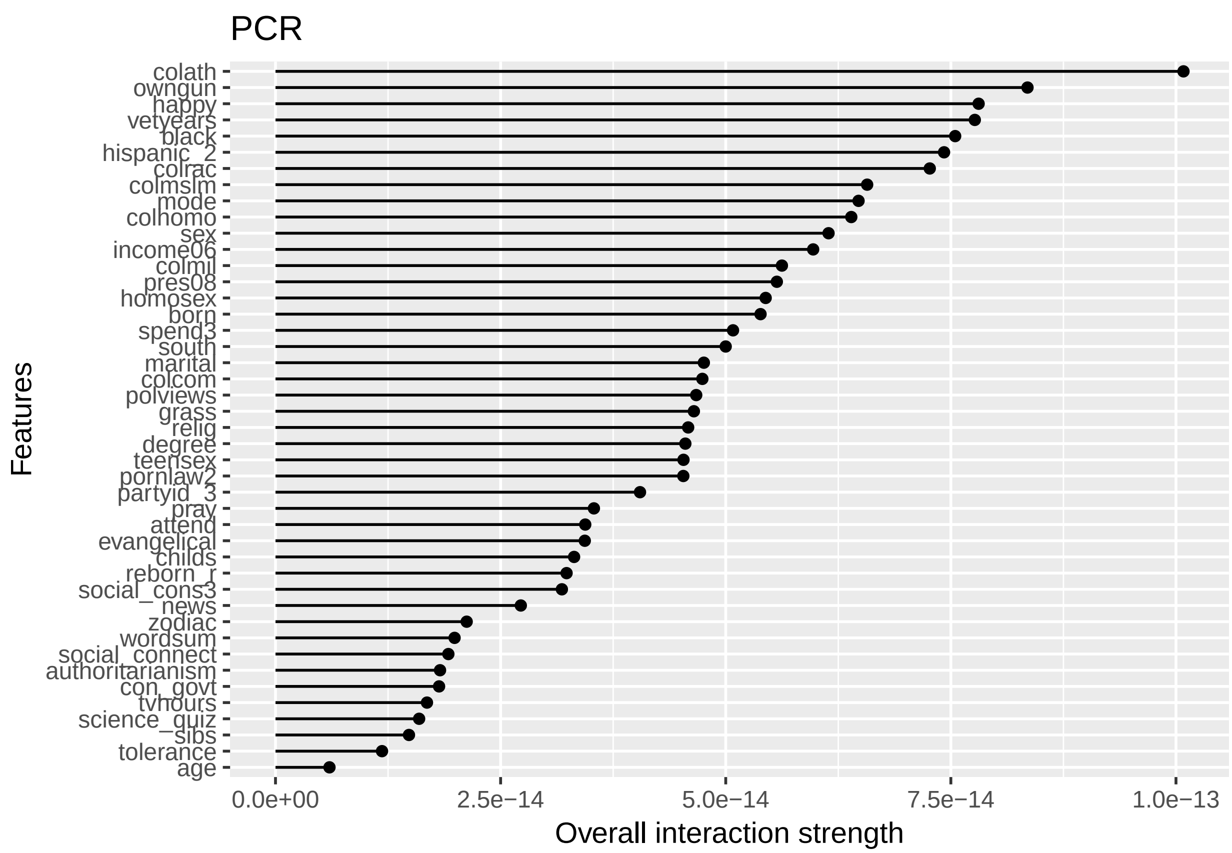
```r
# Elastic Net feature interaction
net_int = Interaction$new(pred_net)
net_int_score = net_int$results
plot(net_int) + ggtitle("Elastic Net")
```




```r
net_int_score %>% arrange(-.interaction) %>% head(5)
```

```
##        .feature .interaction
## 1       colmslm 9.920574e-14
## 2 social_cons3 9.826045e-14
## 3          born 9.528245e-14
## 4        colath 9.329820e-14
## 5       wordsum 9.235250e-14
```

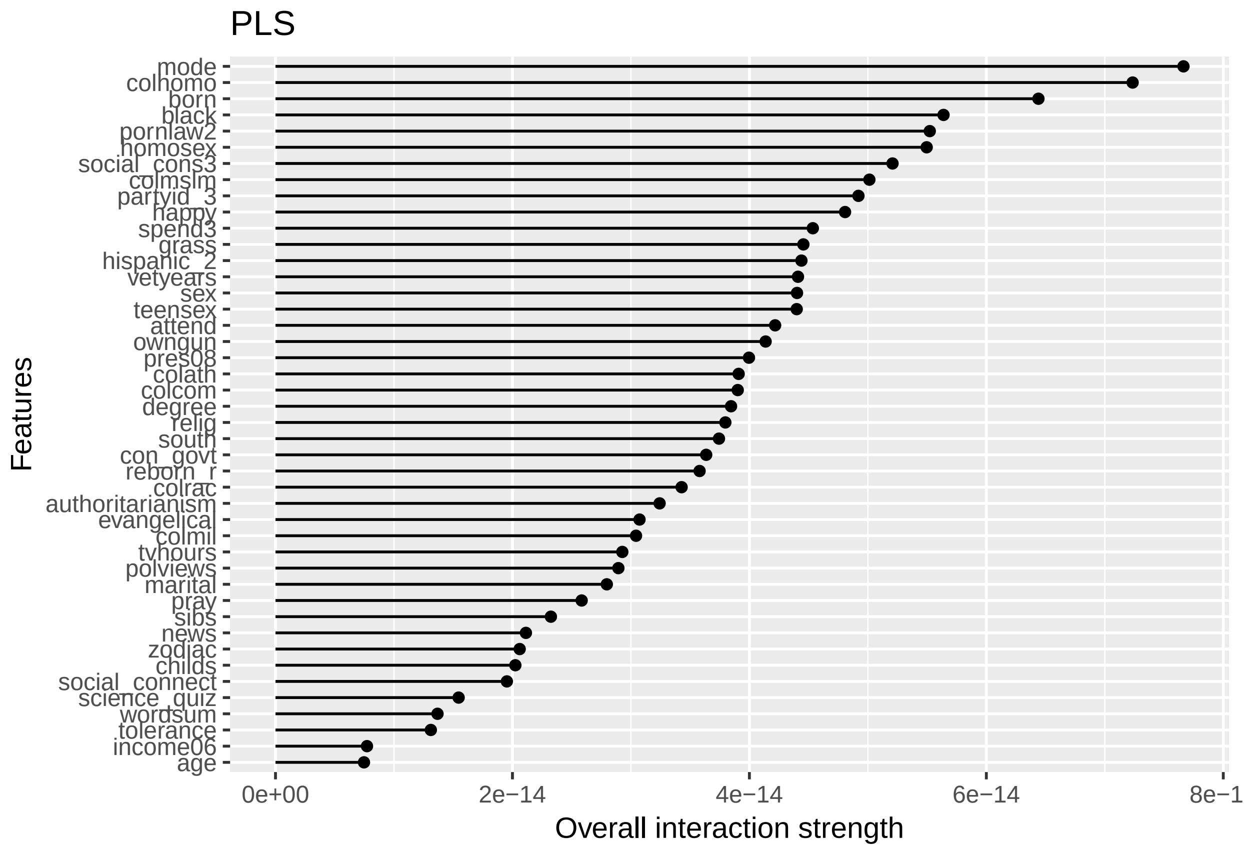
```r
# PCR feature interaction
pcr_int = Interaction$new(pred_pcr)
pcr_int_score = pcr_int$results
plot(pcr_int) + ggtitle("PCR")
```




```r
pcr_int_score %>% arrange(-.interaction) %>% head(5)
```

```
##    .feature .interaction
## 1    colath 1.008344e-13
## 2    owngun 8.351261e-14
## 3     happy 7.808475e-14
## 4 vetyears 7.766055e-14
## 5     black 7.547879e-14
```

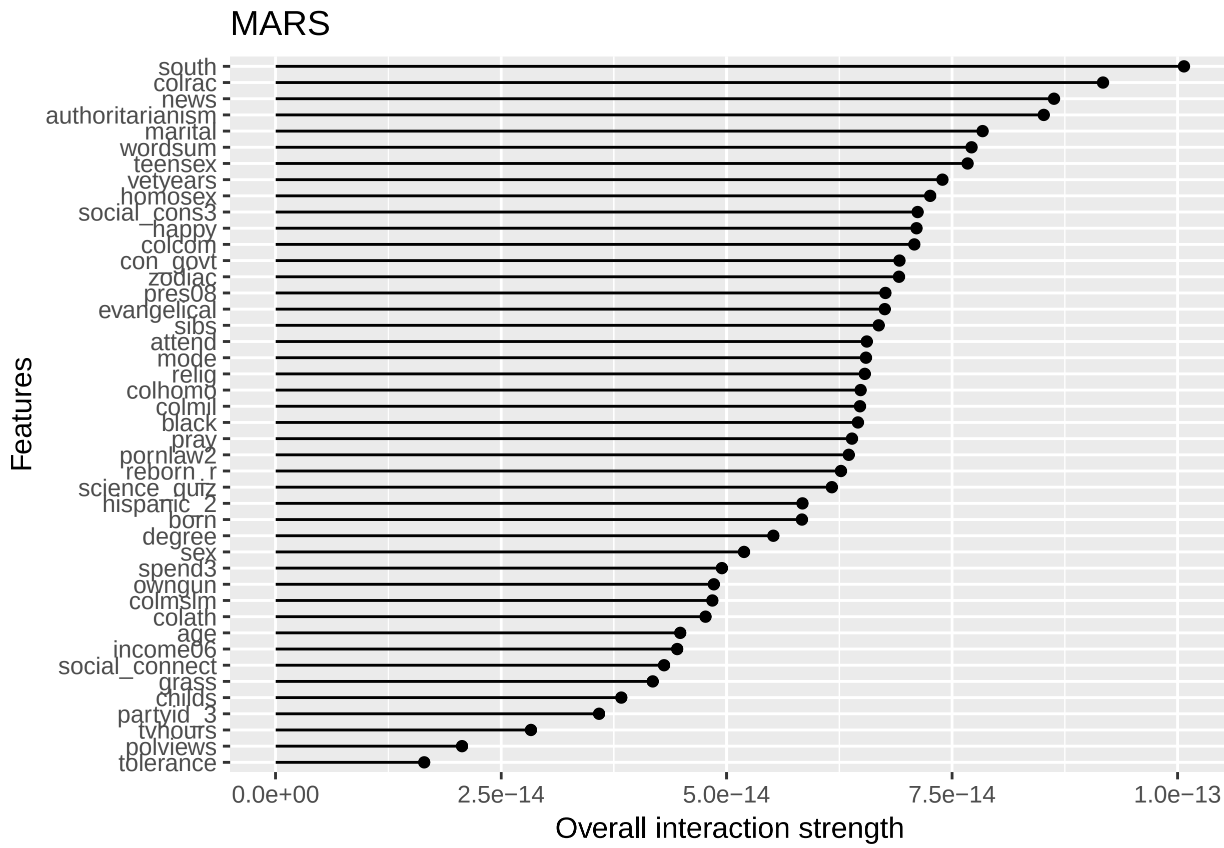
```r
# PLS feature interaction
pls_int = Interaction$new(pred_pls)
pls_int_score = pls_int$results
plot(pls_int) + ggtitle("PLS")
```




```r
pls_int_score %>% arrange(-.interaction) %>% head(5)
```

```
##    .feature .interaction
## 1      mode 7.663972e-14
## 2   colhomo 7.234621e-14
## 3      born 6.439649e-14
## 4     black 5.638743e-14
## 5 pornlaw2 5.522438e-14
```

```r
# MARS feature interaction
mars_int = Interaction$new(pred_mars)
mars_int_score = mars_int$results
plot(mars_int) + ggtitle("MARS")
```




```
mars_int_score %>% arrange(-.interaction) %>% head(5)
```

```
##            .feature .interaction
## 1             south 1.007076e-13
## 2            colrac 9.173213e-14
## 3              news 8.629674e-14
## 4 authoritarianism 8.516293e-14
## 5           marital 7.838219e-14
```

3. Take the optimal model, apply the test set to the model, and calculate the test set MSE. Does this model generalize well to the test set?

```
# I will go on with the Elastic Net model
predicts = predict(ela.net, gss_test)
y_true = gss_test$egalit_scale
mse = sum((y_true - predicts)^2)/length(y_true)
sqrt(mse)
```

```
## [1] 7.84051
```

Generally, the model generalized well to the test set. In the training process, the average CV RMSE is around 7.72, and it does not inflate very much on the test set.