HW4_R

February 16, 2020

[63]:
```
set.seed(123)
library(margins)
library(repr)
library(ggplot2)
library(rsample)
library(dplyr)
library(purrr)
library(splines)
library(msmtools)
options(repr.plot.width=5, repr.plot.height=4)
```

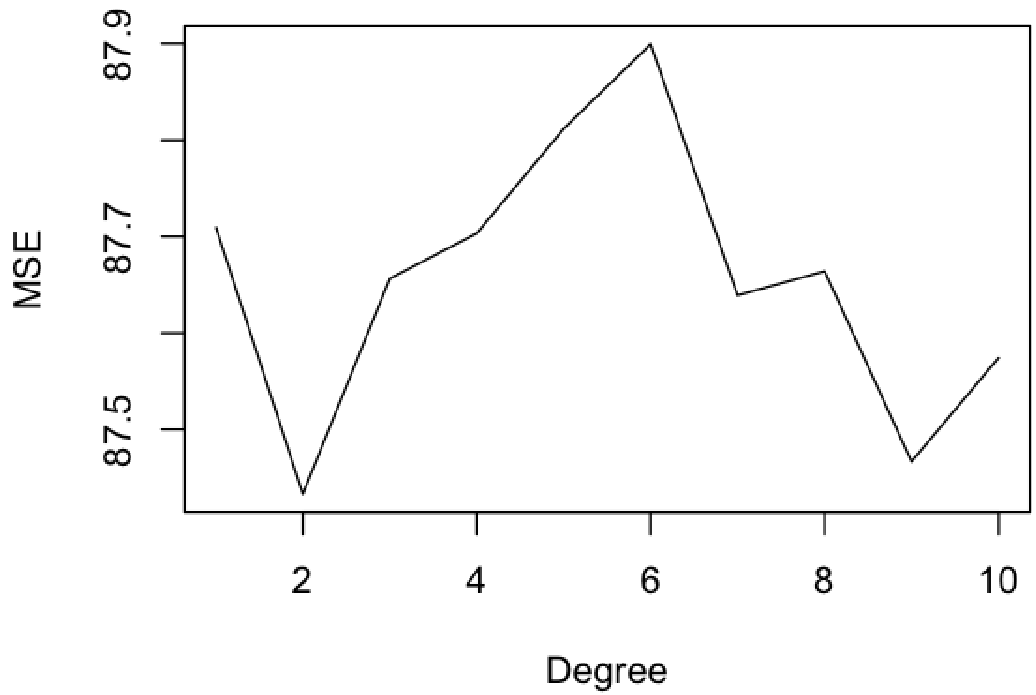
[77]:
```
gss_train = read.csv('data/gss_train.csv')
gss_test = read.csv('data/gss_test.csv')
```

0.1 Egalitarianism and Income

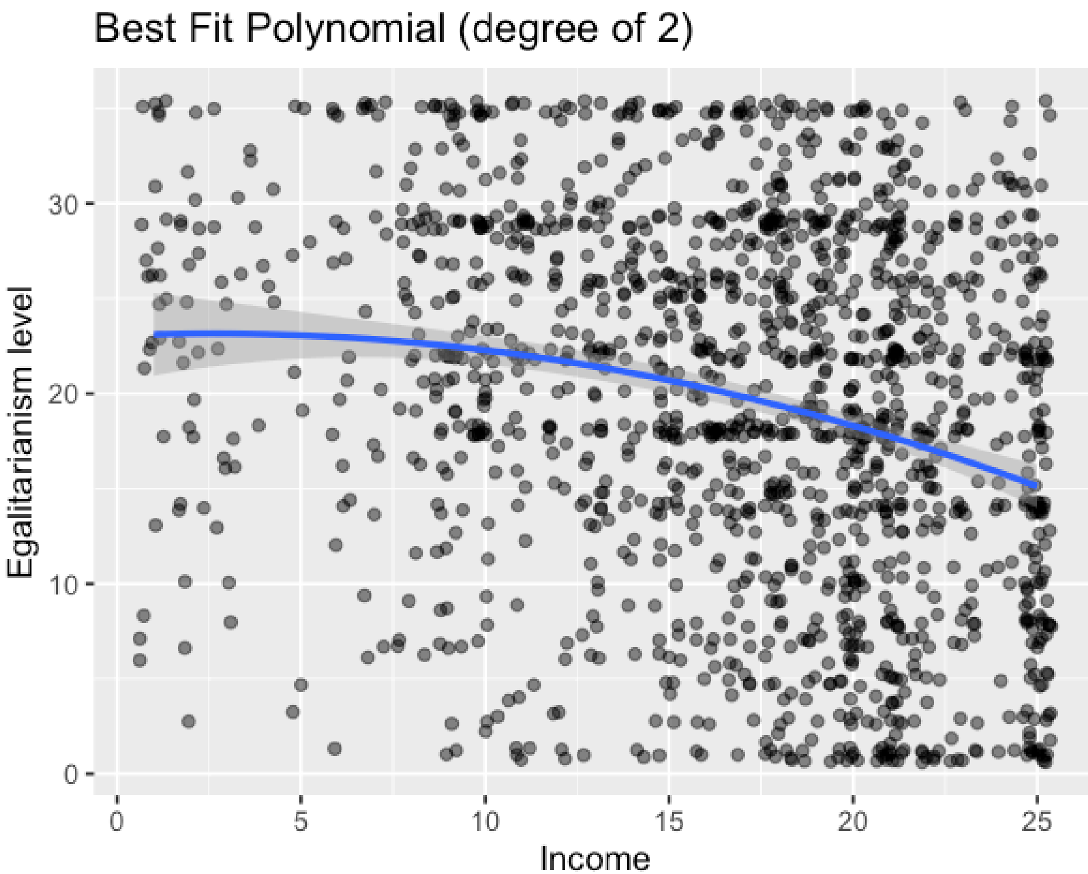
0.1.1 1. Polynomial Regression

[4]:
```
k = 10
fold = sample(k, nrow(gss_train), replace = TRUE)

mse = c()
for(j in 1:k){
    sub_mse = c()
    for(i in 1:k){
        test_set = gss_train[fold==i,]
        train_set = gss_train[fold!=i,]
        model = lm(egalit_scale ~ poly(income06, j), data = train_set)
        pred = predict(model, test_set)
        sub_mse = c(sub_mse, mean((pred - test_set$egalit_scale)^2))
    }
    mse = c(mse, mean(sub_mse))
}
```

[4]:
```
plot(mse, type = 'l', xlab = 'Degree', ylab = 'MSE')
```




```
[133]: ggplot(gss_train, aes(income06, egalit_scale)) +
    geom_jitter(alpha = 0.5) +
    stat_smooth(method = 'lm', formula = y ~ poly(x, degree=2)) +
    labs(title = 'Best Fit Polynomial (degree of 2)',
         x = 'Income', y = 'Egalitarianism level')
```



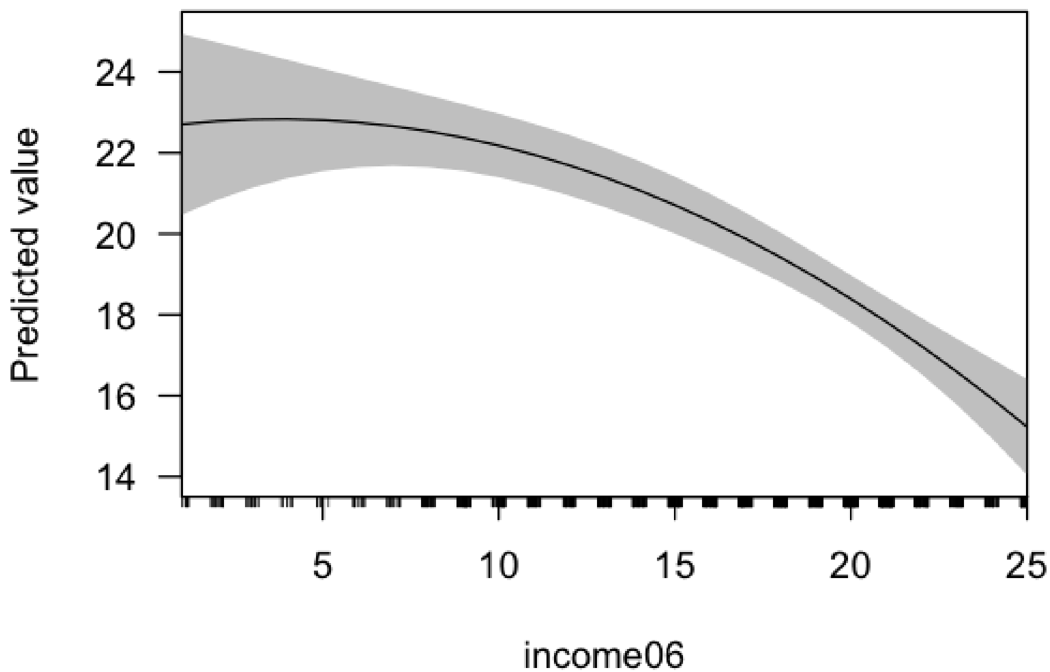
[134]:
```
best_model = lm(egalit_scale ~ stats::poly(income06, 2), data = train_set)
summary(margins(best_model))
```

factor	AME	SE	z	p	lower	upper
income06	-0.4422506	0.04976301	-8.887136	6.270432e-19	-0.5397843	-0.3447169

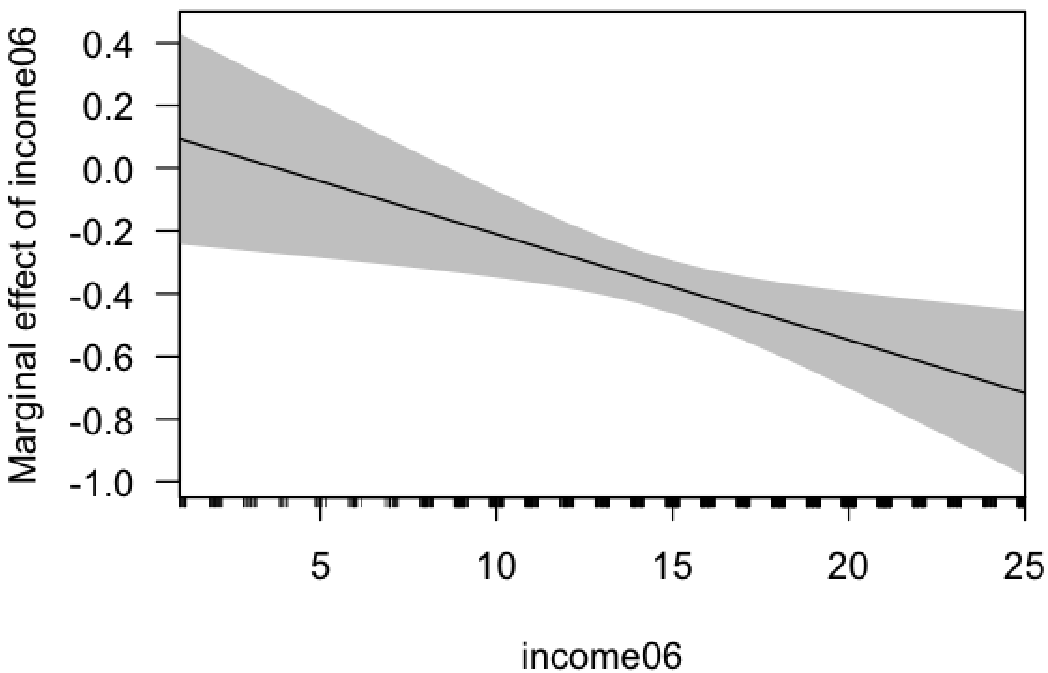
[57]:
```
cplot(best_model, 'income06', what = 'prediction')
```

```
   xvals    yvals    upper    lower
1      1 22.70467 24.93877 20.47057
2      2 22.78141 24.72831 20.83451
3      3 22.82438 24.51346 21.13530
4      4 22.83360 24.29562 21.37159
5      5 22.80906 24.07630 21.54182
6      6 22.75076 23.85695 21.64457
7      7 22.65870 23.63828 21.67911
8      8 22.53288 23.41943 21.64632
9      9 22.37329 23.19701 21.54957
10    10 22.17995 22.96501 21.39489
11    11 21.95285 22.71570 21.19000
```

```
12      12 21.69199 22.44122 20.94276
13      13 21.39737 22.13490 20.65984
14      14 21.06899 21.79188 20.34610
15      15 20.70685 21.40932 20.00439
16      16 20.31095 20.98645 19.63546
17      17 19.88130 20.52486 19.23773
18      18 19.41788 20.02914 18.80661
19      19 18.92070 19.50777 18.33363
20      20 18.38976 18.97336 17.80616
```



[56]:
```
cplot(best_model, 'income06', what = 'effect')
```



As a conclusion, the best polynomial model has 2 degrees, although the increase of MSE is not monotonic as the degree increases. Also, as `income06` increases, its marginal effect decreases from positive to negative. The AME for the model of 2 degrees is -0.4369. This suggests that on average, one unit increase in income decreases the level of egalitarianism by about 0.4 unit. In another word, the richer people tend to be care less about equal distribution of the wealth.

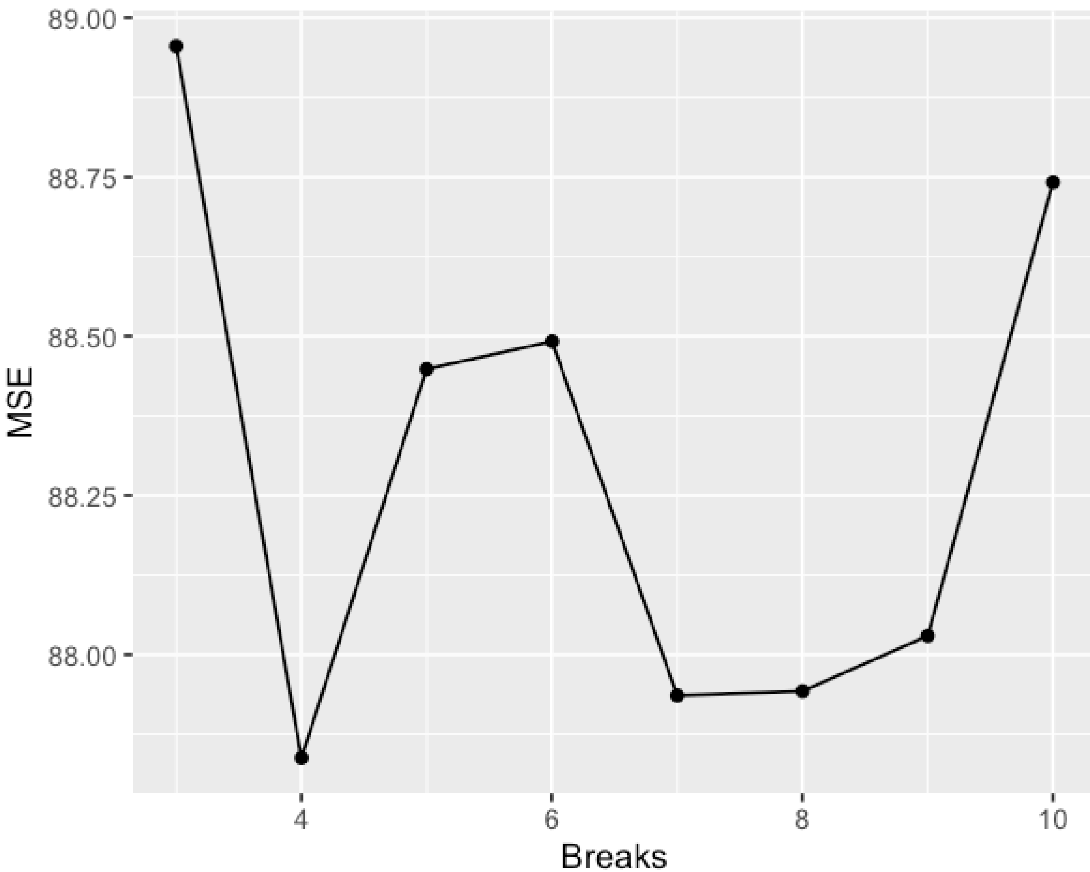
0.1.2 2. Step Function

[14]:
```
mse = c()
for(j in 3:k){
    sub_mse = c()
    for(i in 1:k){
        test_set = gss_train[fold==i,]
        train_set = gss_train[fold!=i,]

        labs = levels(cut(gss_train$income06, j))
        breaks = unique(c(as.numeric(sub("\\((.+),.*", "\\1", labs)),
            as.numeric(sub("[^,]*,([^]]*)\\]", "\\1", labs))))
```

```
        model = lm(egalit_scale~cut(income06,unique(breaks)), data = train_set)
        pred = predict(model, test_set)

        sub_mse = c(sub_mse, mean((pred - test_set$egalit_scale)^2))
    }
    mse = c(mse, mean(sub_mse))
}
```

[15]:

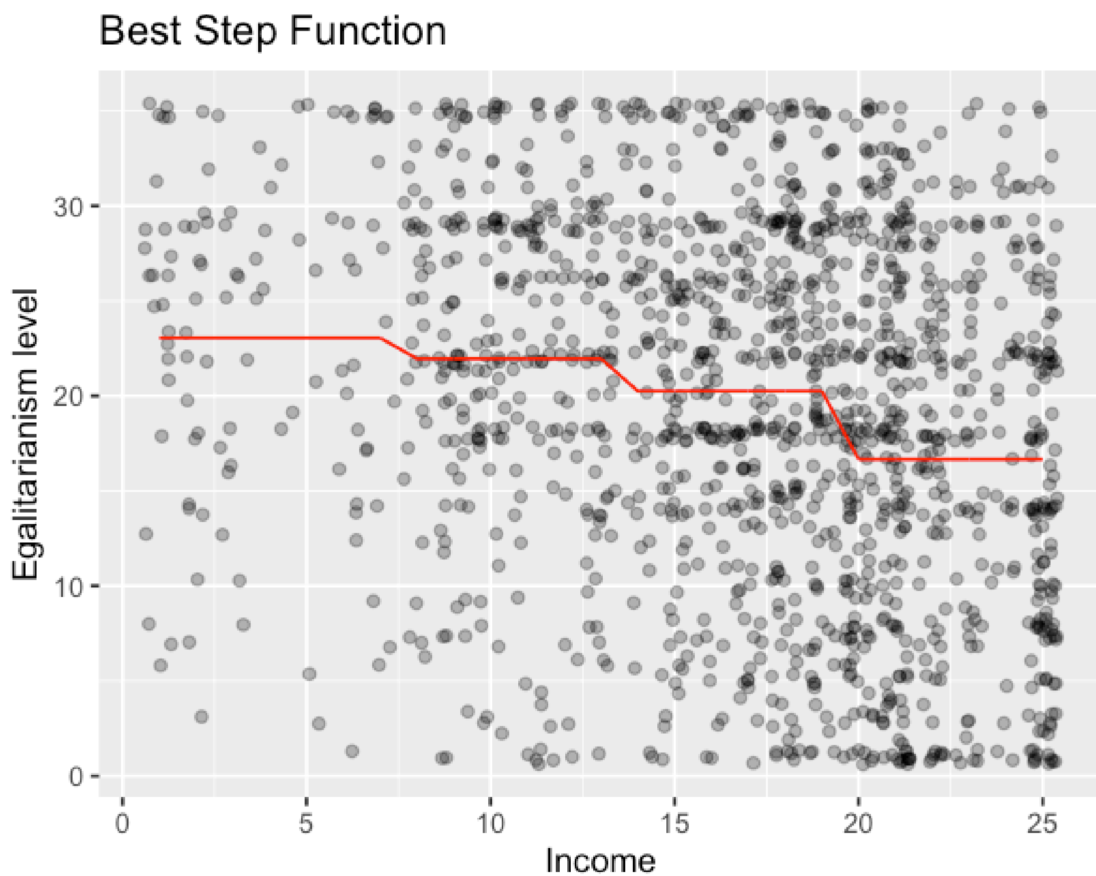
```
step_sum = data.frame('Breaks' = 3:k, "MSE"=mse)
ggplot(step_sum, aes(Breaks, MSE))+ geom_line()+geom_point()
```



[50]:

```
labs = levels(cut(gss_train$income06, 4))
breaks = unique(c(as.numeric(sub("\\((.+),.*", "\\1", labs)), as.
↪numeric(sub("[^,]*,([^]]*)\\]", "\\1", labs))))
best_model = lm(egalit_scale~cut(income06,unique(breaks)), data = gss_train)
pred = predict(best_model, gss_train)
df_pred = data.frame('income06' = gss_train$income06, 'pred' = pred)
```

[54]:
```
ggplot(gss_train, aes(income06, egalit_scale)) +
    geom_jitter(alpha = 0.3) +
    geom_line(data = df_pred, aes(income06, pred), color = 'red') +
    labs(title = 'Best Step Function',
         x = 'Income', y = 'Egalitarianism level')
```



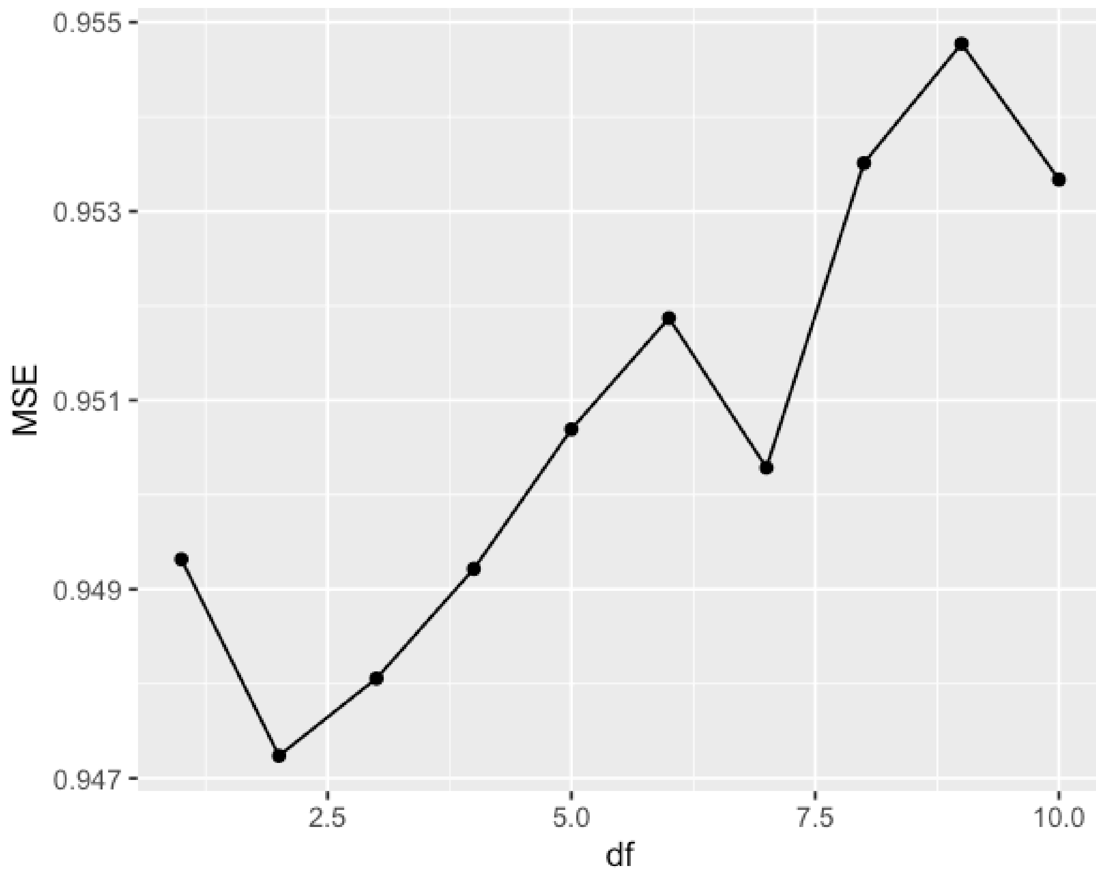
As a summary, the best step function has 4 breaks (5 intervals). Still, as income increases, the predicted level of egalitarianism decreases step by step.

0.1.3 Natural Regression Spline

[59]:
```
mse = c()
for(j in 1:k){
    sub_mse = c()
    for(i in 1:k){
        test_set = gss_train[fold==i,]
        train_set = gss_train[fold!=i,]
        model = lm(egalit_scale ~ ns(income06, df = j), data = train_set)
```

```
        pred = predict(model, test_set)
        sub_mse = c(sub_mse, mean((pred - test_set$egalit_scale)^2))
    }
    mse = c(mse, mean(sub_mse))
}
```

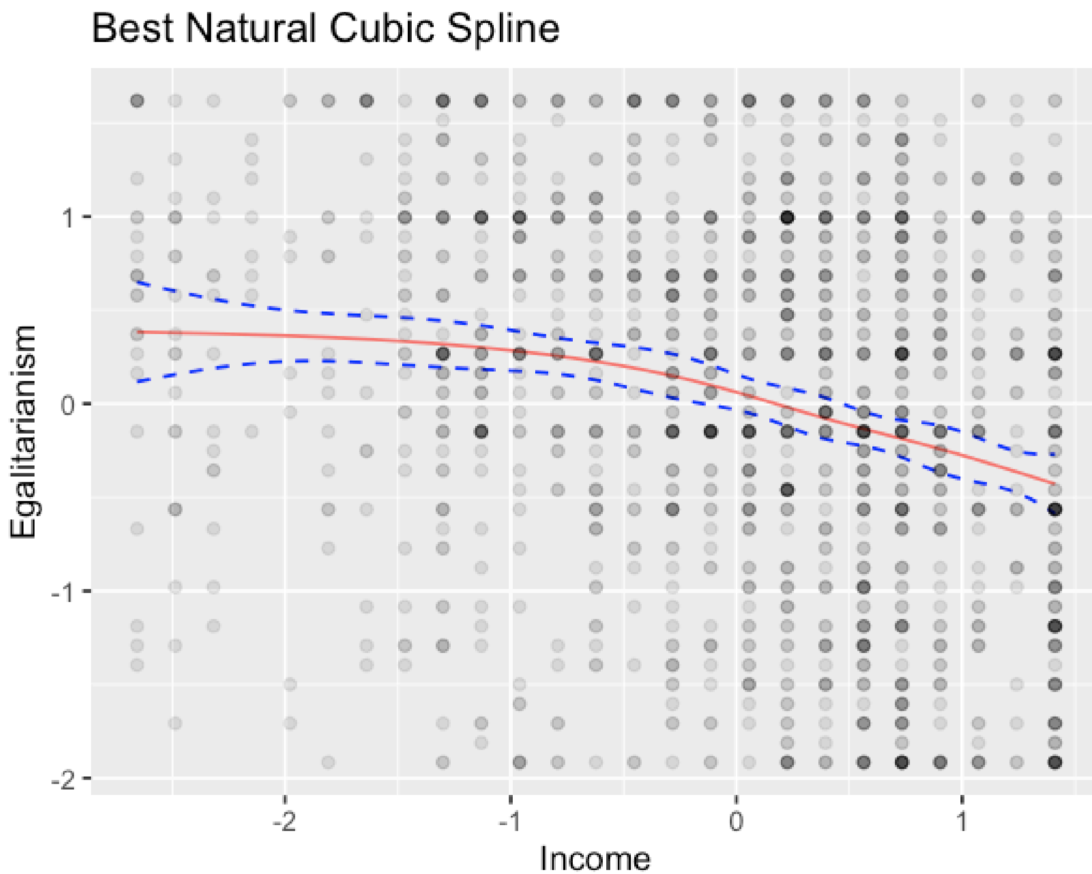
[60]:
```
step_sum = data.frame('df' = 1:k, "MSE"=mse)
ggplot(step_sum, aes(df, MSE))+ geom_line()+geom_point()
## best knots is 2 (df = 5)
```



[62]:
```
model = glm(egalit_scale ~ ns(income06, df = 5), data = gss_train) %>%
        cplot("income06", what = "prediction", n = 100, draw = FALSE)
model %>% ggplot(aes(x = xvals)) +
        geom_line(aes(y = yvals, color = 'red')) +
        geom_line(aes(y = upper), linetype = 2, color = 'blue') +
        geom_line(aes(y = lower), linetype = 2, color = 'blue') +
        geom_point(data = gss_train, aes(income06, egalit_scale), alpha = 0.1) +
        labs(x = "Income", y = "Egalitarianism", title = "Best Natural Cubic␣
 ↪Spline") +
```

```r
    theme(legend.position = "none")
```

```
       xvals     yvals     upper     lower
1  -2.655726 0.3829889 0.6488579 0.1171198
2  -2.614648 0.3819858 0.6370112 0.1269603
3  -2.573571 0.3809774 0.6253548 0.1366000
4  -2.532494 0.3799584 0.6139188 0.1459980
5  -2.491416 0.3789235 0.6027349 0.1551122
6  -2.450339 0.3778675 0.5918363 0.1638987
7  -2.409262 0.3767850 0.5812575 0.1723125
8  -2.368185 0.3756707 0.5710341 0.1803073
9  -2.327107 0.3745193 0.5612028 0.1878359
10 -2.286030 0.3733256 0.5518001 0.1948512
11 -2.244953 0.3720843 0.5428621 0.2013064
12 -2.203875 0.3707900 0.5344232 0.2071567
13 -2.162798 0.3694374 0.5265143 0.2123605
14 -2.121721 0.3680213 0.5191614 0.2168813
15 -2.080644 0.3665364 0.5123831 0.2206897
16 -2.039566 0.3649774 0.5061890 0.2237658
17 -1.998489 0.3633389 0.5005770 0.2261009
18 -1.957412 0.3616158 0.4955323 0.2276992
19 -1.916334 0.3598026 0.4910261 0.2285790
20 -1.875257 0.3578941 0.4870157 0.2287725
```



As shown from the two figures above, the best natural regression spline has 2 knots (df = 3). This model is smoother compared to the previous two models and basically demonstrates the same information that, as income increases, people tend to be less egalitarian.

0.1.4 4. Egalitarianism and Everything

[44]:
```
options(warn=-1)
```

Data Pre-Processing see reference https://stackoverflow.com/questions/23619188/r-scaling-numeric-values-only-in-a-dataframe-with-mixed-types/36819351#36819351

[81]:
```
standardize = function(data){
    df = data %>%
    mutate_if(is.matrix, as.numeric) %>%
    mutate_if(is.numeric, c) %>%
    mutate_if(is.numeric, scale)
```

```r
a = as.character(df$relig)
a[a %in% c('CATHOLIC', 'PROTESTANT', 'CHRISTIAN')] = 1
a[!(a %in% c('CATHOLIC', 'PROTESTANT', 'CHRISTIAN'))] = 0
df$relig = as.numeric(a)

a = as.character(df$marital)
a[a == 'Married'] = 1
a[a != 'Married'] = 0
df$marital = as.numeric(a)

a = as.character(df$attend)
a[a == 'Never'] = 0
a[(a == '<Once/yr')|(a == 'Once/yr')] = 1
a[a == 'Sev times/yr'] = 2
a[a == 'Once/mo'] = 3
a[a == '2-3 times /mo'] = 4
a[(a == 'Every wk')|(a =='>Once/wk')|(a == 'Nrly evry wk')] = 5
df$attend = as.numeric(a)

a = as.character(df$polviews)
a[a == 'ExtrmLib'] = 0
a[a == 'Liberal'] = 1
a[a == 'SlghtLib'] = 2
a[a == 'Moderate'] = 3
a[a == 'SlghtCons'] = 4
a[a == 'Conserv'] = 5
a[a == 'ExtrmCons'] = 6
df$polviews = as.numeric(a)


a = as.character(df$degree)
a[a == '<HS'] = 0
a[a == 'HS'] = 1
a[a == 'Junior Coll'] = 2
a[a == 'Bachelor deg'] = 3
a[a == 'Graduate deg'] = 4
df$degree = as.numeric(a)

a = as.character(df$news)
a[a == 'NEVER'] = 0
a[a == 'LESS THAN ONCE WK'] = 1
a[a == 'ONCE A WEEK'] = 2
a[a == 'FEW TIMES A WEEK'] = 3
a[a == 'EVERYDAY'] = 4
df$news = as.numeric(a)
```

```
    a = as.character(df$pray)
    a[a == 'NEVER'] = 0
    a[a == 'ONCE A WEEK'] = 1
    a[a == 'LT ONCE A WEEK'] = 2
    a[a == 'SEVERAL TIMES A WEEK'] = 3
    a[a == 'ONCE A DAY'] = 4
    a[a == 'SEVERAL TIMES A DAY'] = 5
    df$pray = as.numeric(a)

    df
}

gss_train = standardize(gss_train)
gss_test = standardize(gss_test)
```

0.1.5 Linear Regression

There is no need to tune anything for linear regression. The average of 10-fold CV MSE is 0.6623, and the whole model MSE is 0.8918. When all predictors are considered, egalitarianism is siginificantly related to political views: the more liberal, the more egalitarinian. Younger and poorer people, aside from political views, are also more egalitarinian.

[42]:
```
library(caret)
```

[45]:
```
folds = trainControl(method = "cv", number = 10)

gss_lm = train(
    egalit_scale ~ ., data = gss_train, method = "lm",
    trControl = folds,
    tuneLength = 10)
gss_lm
```

```
Linear Regression

1481 samples
  44 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1334, 1333, 1332, 1333, 1334, 1334, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8137494  0.3448641  0.6456499

Tuning parameter 'intercept' was held constant at a value of TRUE
```

[46]:
```
gss_lm_test = train(
    egalit_scale ~ ., data = gss_test, method = "lm")
gss_lm_test
```

```
Linear Regression

493 samples
 44 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 493, 493, 493, 493, 493, 493, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.9298948  0.2100389  0.7352836

Tuning parameter 'intercept' was held constant at a value of TRUE
```

[47]:
```
0.8138173**2
0.9443879**2
```

0.66229859777929

0.89186850566641

[48]:
```
model = lm(egalit_scale ~ ., data = gss_train)
pred = predict(model, gss_test)
print(mean((pred - gss_test$egalit_scale)^2))
summary(model)
```

```
[1] 0.7035309

Call:
lm(formula = egalit_scale ~ ., data = gss_train)

Residuals:
     Min       1Q   Median       3Q      Max
-2.91394 -0.50116 -0.00216  0.53007  2.11940

Coefficients: (2 not defined because of singularities)
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                0.7613198  0.2981538   2.553  0.01077 *
age                       -0.1123108  0.0277383  -4.049 5.42e-05 ***
attend                    -0.0012061  0.0133492  -0.090  0.92802
authoritarianism           0.0046642  0.0247906   0.188  0.85079
blackYes                   0.0995314  0.0717092   1.388  0.16536
bornYES                   -0.0310722  0.0701998  -0.443  0.65810
```

childs	0.0545790	0.0246579	2.213	0.02703	*
colathNOT ALLOWED	0.0476824	0.0618851	0.770	0.44113	
colracNOT ALLOWED	-0.0028243	0.0559383	-0.050	0.95974	
colcomNOT FIRED	-0.0012327	0.0569077	-0.022	0.98272	
colmilNOT ALLOWED	-0.1218429	0.0582975	-2.090	0.03679	*
colhomoNOT ALLOWED	0.0922454	0.0802521	1.149	0.25057	
colmslmYes, allowed	-0.0185229	0.0579924	-0.319	0.74947	
con_govt	-0.0333960	0.0226076	-1.477	0.13984	
degree	-0.0425028	0.0227422	-1.869	0.06184	.
evangelicalLow	0.0962189	0.1153185	0.834	0.40421	
evangelicalMod	0.0786875	0.0815192	0.965	0.33458	
grassNOT LEGAL	-0.2487933	0.0566247	-4.394	1.20e-05	***
happyPRETTY HAPPY	-0.0515772	0.0636449	-0.810	0.41785	
happyVERY HAPPY	-0.1059799	0.0707509	-1.498	0.13437	
hispanic_2Yes	0.0035464	0.0728445	0.049	0.96118	
homosexALWAYS WRONG	0.0088042	0.1589927	0.055	0.95585	
homosexNOT WRONG AT ALL	-0.0238809	0.1598742	-0.149	0.88128	
homosexSOMETIMES WRONG	0.0002982	0.1796307	0.002	0.99868	
income06	-0.0652442	0.0255296	-2.556	0.01070	*
marital	NA	NA	NA	NA	
modeOVER THE PHONE	0.0361827	0.0647565	0.559	0.57642	
news	-0.0148825	0.0146332	-1.017	0.30931	
owngunREFUSED	-0.0593114	0.1917475	-0.309	0.75712	
owngunYES	-0.0838512	0.0515163	-1.628	0.10382	
partyid_3Ind	-0.2238890	0.0538095	-4.161	3.36e-05	***
partyid_3Rep	-0.3762475	0.0790325	-4.761	2.13e-06	***
polviews	-0.1754156	0.0180892	-9.697	< 2e-16	***
pornlaw2Not illegal to all	-0.0633641	0.0609583	-1.039	0.29877	
pray	-0.0108180	0.0158312	-0.683	0.49451	
pres08Obama	0.4182243	0.0671503	6.228	6.21e-10	***
reborn_rYes	0.0589227	0.0957148	0.616	0.53825	
relig	NA	NA	NA	NA	
science_quiz	-0.0140464	0.0283595	-0.495	0.62047	
sexMale	-0.1375263	0.0498760	-2.757	0.00590	**
sibs	0.0458757	0.0232703	1.971	0.04887	*
social_connect	0.0029566	0.0230152	0.128	0.89780	
social_cons3Liberal	-0.0288207	0.0803269	-0.359	0.71980	
social_cons3Mod	0.0322244	0.0631606	0.510	0.60999	
southSouth	-0.0366951	0.0475652	-0.771	0.44056	
spend3Liberal	0.1792603	0.0514608	3.483	0.00051	***
spend3Mod	0.0781263	0.0568628	1.374	0.16968	
teensexALWAYS WRONG	0.0578139	0.0685319	0.844	0.39903	
teensexNOT WRONG AT ALL	0.0116106	0.1143490	0.102	0.91914	
teensexSOMETIMES WRONG	0.0487270	0.0890484	0.547	0.58433	
tolerance	-0.0930304	0.0376209	-2.473	0.01352	*
tvhours	0.0511016	0.0229663	2.225	0.02623	*
vetyearsLESS THAN 2 YRS	0.0875443	0.1593179	0.549	0.58275	
vetyearsMORE THAN 4 YRS	0.0572697	0.1448891	0.395	0.69271	

```
vetyearsNONE                0.0706486  0.0987147   0.716  0.47430
wordsum                     0.0133033  0.0262077   0.508  0.61181
zodiacARIES                -0.1760371  0.1055921  -1.667  0.09571 .
zodiacCANCER               -0.0418149  0.1053861  -0.397  0.69159
zodiacCAPRICORN            -0.0562682  0.1039193  -0.541  0.58828
zodiacGEMINI               -0.1443622  0.1006603  -1.434  0.15175
zodiacLEO                  -0.0848368  0.0974235  -0.871  0.38401
zodiacLIBRA                -0.1235218  0.0996629  -1.239  0.21540
zodiacPISCES               -0.1420679  0.1046778  -1.357  0.17494
zodiacSAGITTARIUS          -0.1221534  0.1035906  -1.179  0.23852
zodiacSCORPIO              -0.1733468  0.1044019  -1.660  0.09706 .
zodiacTAURUS               -0.0160679  0.1007930  -0.159  0.87336
zodiacVIRGO                 0.0197235  0.1038762   0.190  0.84943
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7944 on 1416 degrees of freedom
Multiple R-squared:  0.3962,Adjusted R-squared:  0.3689
F-statistic: 14.52 on 64 and 1416 DF,  p-value: < 2.2e-16
```

0.1.6 Elastic Net Regression

With 10 fold elastic net, the best alpha is 0.6 and the best lambda is 0.03345934. The train CV MSE is 0.68 and the final MSE is 0.74. This is much better than the linear model.

[49]:
```
folds = trainControl(method = "cv", number = 10)
gss_elnet = train(
    egalit_scale ~ ., data = gss_train, method = "glmnet",
    trControl = folds,
    tuneLength = 10
)
best_alpha = gss_elnet$bestTune$alpha
best_lambda = gss_elnet$bestTune$lambda
```

[50]:
```
elnet_best = train(egalit_scale ~ ., data = gss_test,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = best_alpha,
                                          lambda = best_lambda))
elnet_best
```

```
glmnet

493 samples
 44 predictor
```

```
No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 493, 493, 493, 493, 493, 493, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8589117  0.2740568  0.6873249

Tuning parameter 'alpha' was held constant at a value of 0.7
Tuning
 parameter 'lambda' was held constant at a value of 0.03345934
```

[51]:
```
mean(gss_elnet$results$RMSE^2)
elnet_best$results$RMSE^2
```

0.676020168272908

0.737729330083205

0.1.7 Principal Component Regression

The best number of components is 10, the train CV MSE is 0.78 and the final MSE is 0.78. This means that PCR performs slightly worse than the elastic net model because our dataset has lots of categorical variables. But it is still better than the linear model because the linear model does not consider the covariance between predictors.

[52]:
```
folds = trainControl(method = "cv", number = 10)
gss_pcr = train(
    egalit_scale ~ ., data = gss_train, method = "pcr",
    trControl = folds,
    tuneLength = 10)
```

[53]:
```
gss_pcr
```

```
Principal Component Analysis

1481 samples
  44 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1334, 1333, 1333, 1332, 1334, 1334, ...
Resampling results across tuning parameters:

  ncomp  RMSE       Rsquared    MAE
   1     0.9897603  0.02926820  0.8241770
   2     0.9684009  0.06483337  0.8079843
```

```
  3     0.9707907  0.06163400  0.8084389
  4     0.8410155  0.29335015  0.6855206
  5     0.8406428  0.29384154  0.6841887
  6     0.8411283  0.29293655  0.6846652
  7     0.8416448  0.29208661  0.6850168
  8     0.8421907  0.29111912  0.6848049
  9     0.8424655  0.29066026  0.6851048
 10     0.8422671  0.29084255  0.6854361

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was ncomp = 5.
```

[54]:
```
pcr_best = train(
    egalit_scale ~ ., data = gss_test, method = "pcr",
    tuneGrid = expand.grid(ncomp = gss_pcr$bestTune$ncomp))
pcr_best
```

```
Principal Component Analysis

493 samples
 44 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 493, 493, 493, 493, 493, 493, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8826469  0.2142523  0.7196249

Tuning parameter 'ncomp' was held constant at a value of 5
```

[55]:
```
mean(gss_pcr$results$RMSE^2)
pcr_best$results$RMSE^2
```

0.78181575682232

0.779065551997975

0.1.8 Partial least squares regression

The best number of components is 6, the train CV MSE is 0.67 and the final MSE is 0.79. This means that PLS performs worse than the previous two models but better than the linear model.

[56]:
```
folds = trainControl(method = "cv", number = 10)
gss_pls = train(
```

```
    egalit_scale ~ ., data = gss_train, method = "pls",
    trControl = folds,
    tuneLength = 10)
gss_pls
```

```
Partial Least Squares

1481 samples
  44 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1333, 1333, 1332, 1332, 1333, 1334, ...
Resampling results across tuning parameters:

  ncomp  RMSE       Rsquared   MAE
   1     0.8681898  0.2474046  0.7087913
   2     0.8308068  0.3122735  0.6740116
   3     0.8229471  0.3256703  0.6634647
   4     0.8127777  0.3437296  0.6462019
   5     0.8047999  0.3566256  0.6395816
   6     0.8059920  0.3548967  0.6386655
   7     0.8082478  0.3514184  0.6397831
   8     0.8088984  0.3505123  0.6392299
   9     0.8096682  0.3493316  0.6402650
  10     0.8104453  0.3481965  0.6412029

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was ncomp = 5.
```

[57]:
```
pls_best = train(
    egalit_scale ~ ., data = gss_test, method = "pls",
    tuneGrid = expand.grid(ncomp = gss_pls$bestTune$ncomp))
pls_best
```

```
Partial Least Squares

493 samples
 44 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 493, 493, 493, 493, 493, 493, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8884938  0.2415459  0.7077435
```

```
Tuning parameter 'ncomp' was held constant at a value of 5
```

[58]:
```
mean(gss_pls$results$RMSE^2)
pls_best$results$RMSE^2
```

0.669913423225353

0.789421254989816

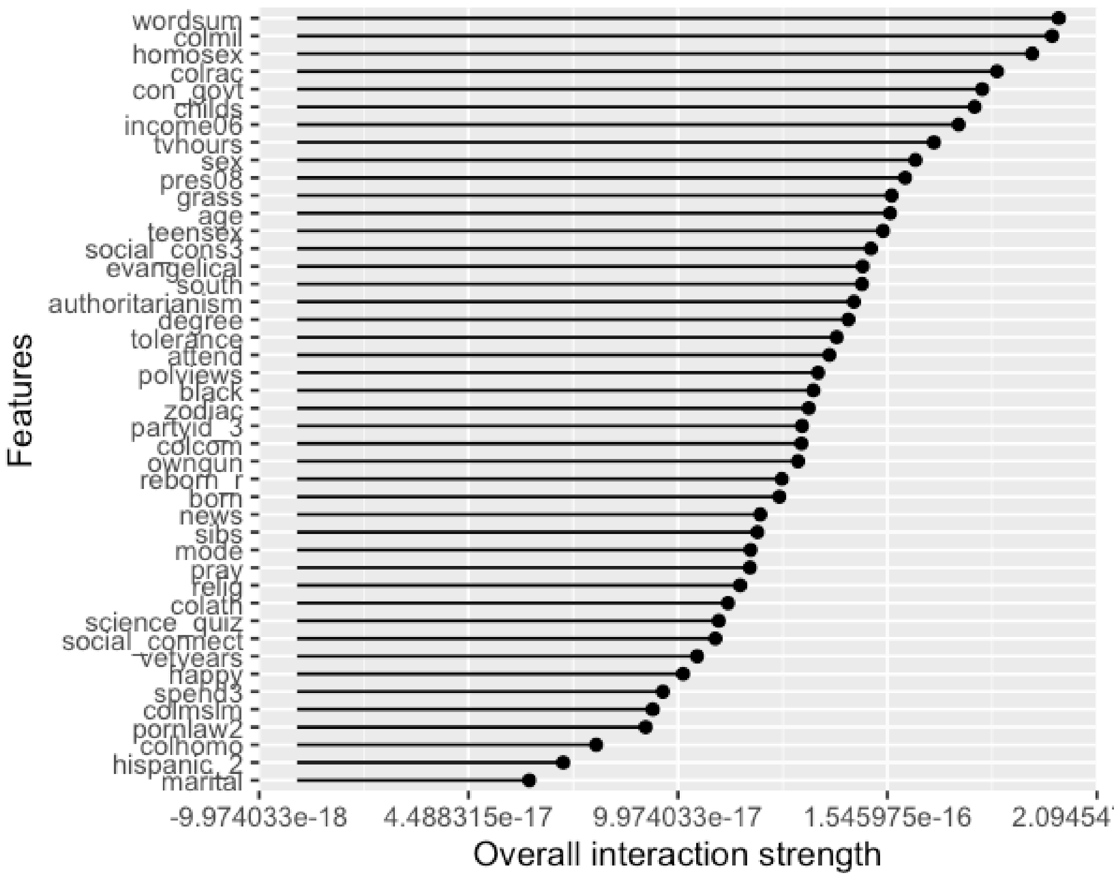
0.1.9 5. Feature Interaction Plots

[66]:
```
library("iml")
#library('vip')
```

[71]:
```
features = gss_train %>% dplyr::select(-egalit_scale)
response = gss_train$egalit_scale
```

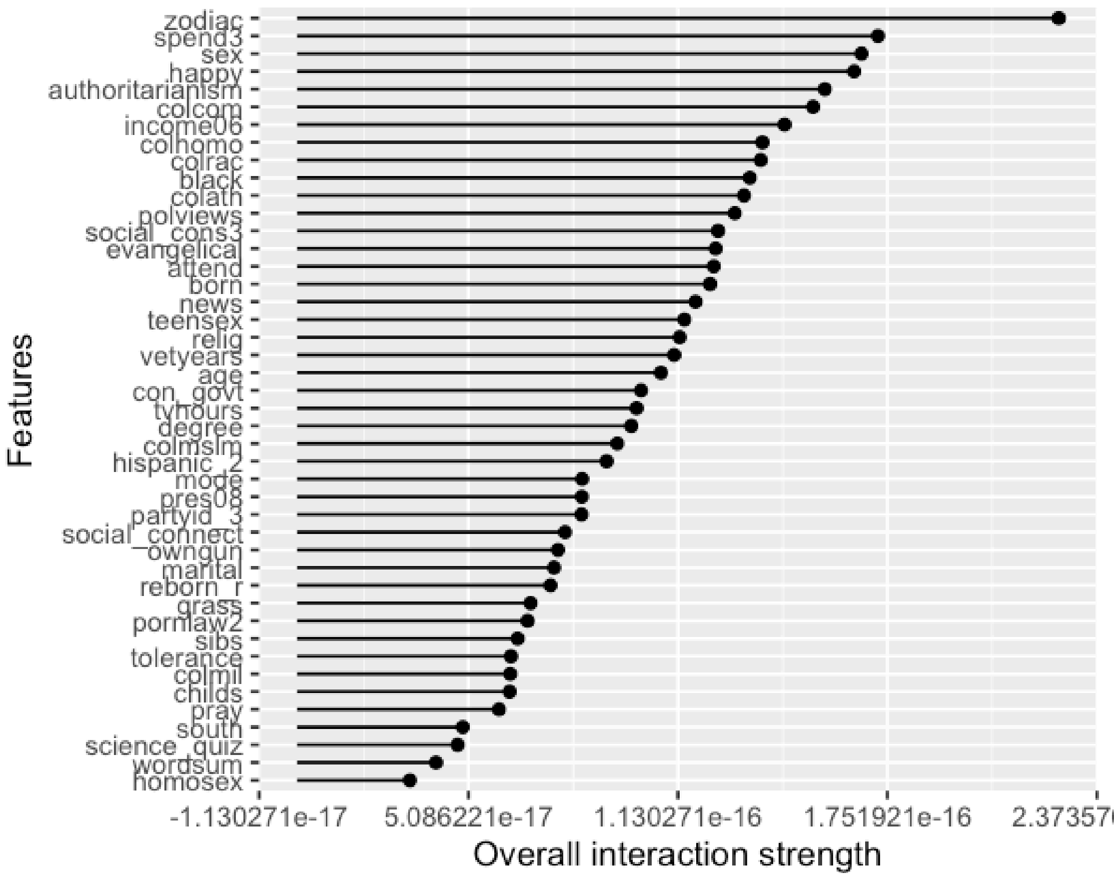
[72]:
```
predictor_lm = Predictor$new(model = gss_lm,
                             data = features,
                             y = response)
```

[70]:
```
plot(Interaction$new(predictor_lm))
```



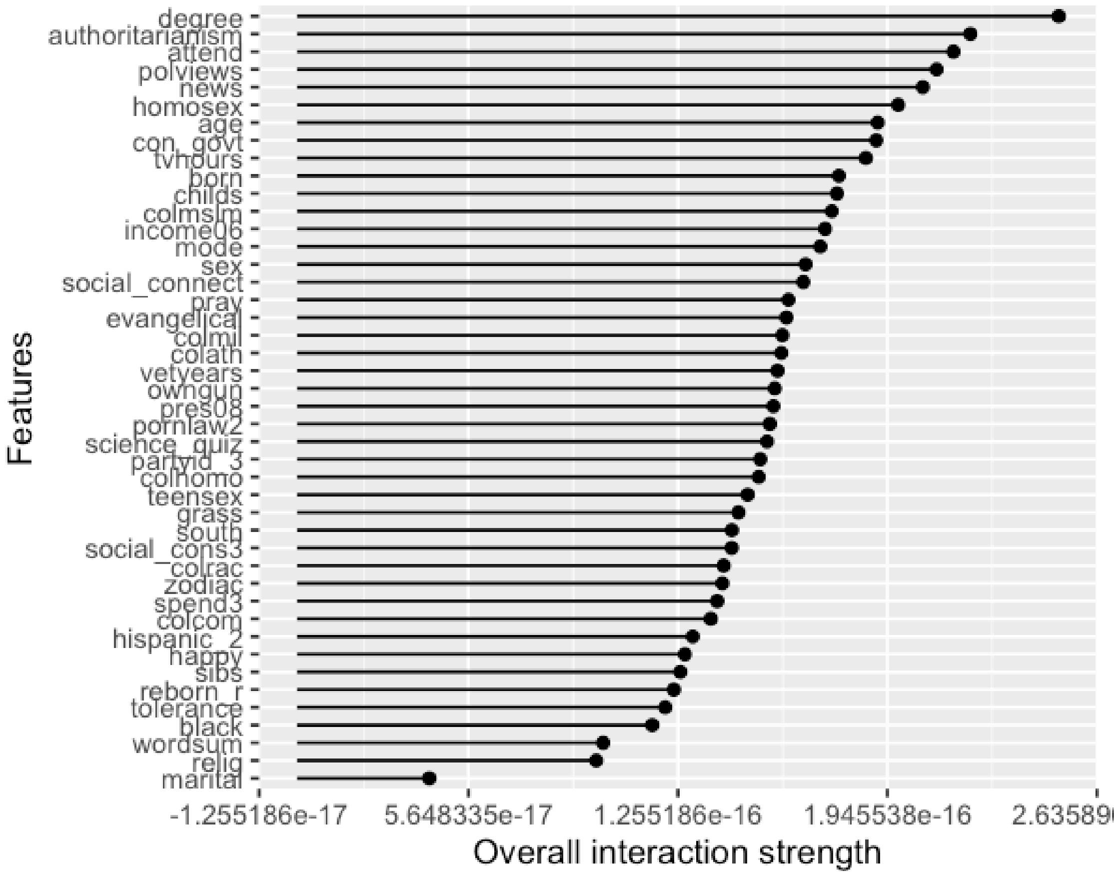
[73]:

```
predictor_elnet = Predictor$new(model = elnet_best,
                                data = features,
                                y = response)
plot(Interaction$new(predictor_elnet))
```

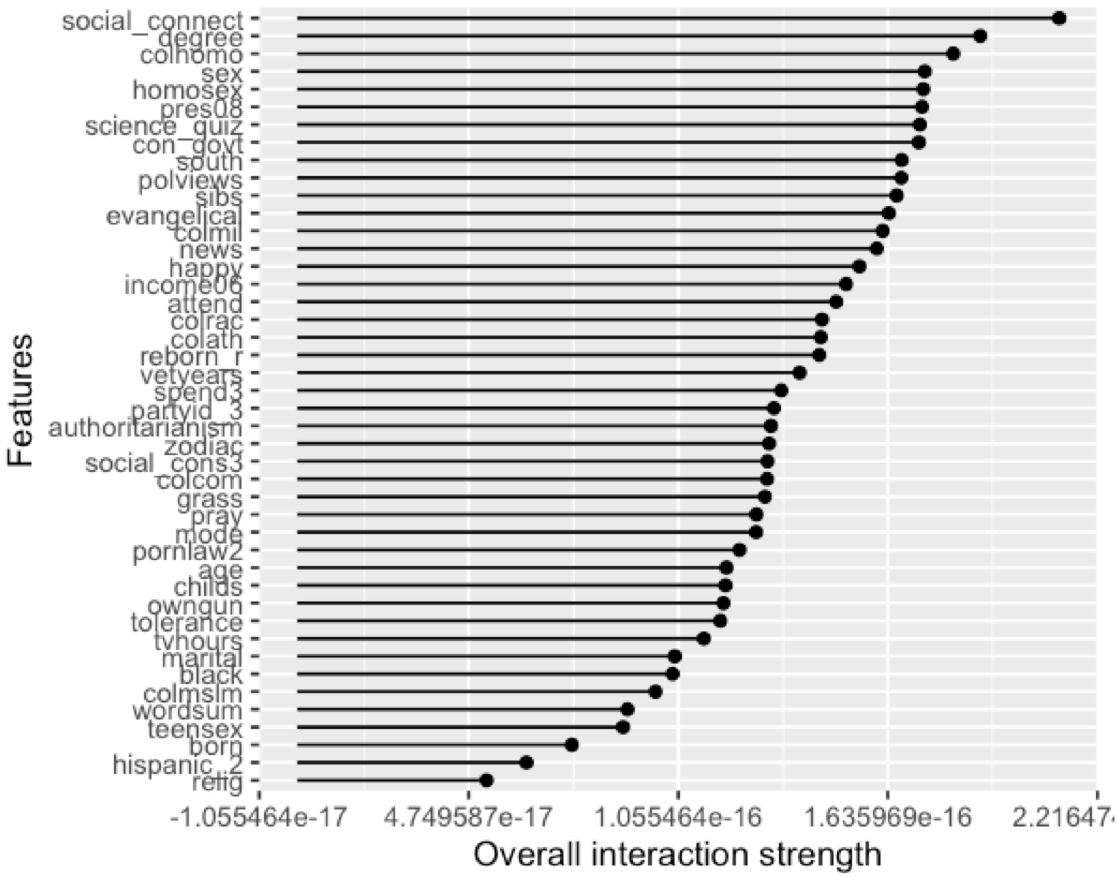


[74]:

```
predictor_pcr = Predictor$new(model = pcr_best,
                              data = features,
                              y = response)
plot(Interaction$new(predictor_pcr))
```



[75]:
```
predictor_pls = Predictor$new(model = pls_best,
                              data = features,
                              y = response)
plot(Interaction$new(predictor_pls))
```



[85]:
```
elnet_best
```

```
glmnet

493 samples
 44 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 493, 493, 493, 493, 493, 493, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8589117  0.2740568  0.6873249

Tuning parameter 'alpha' was held constant at a value of 0.7
Tuning
 parameter 'lambda' was held constant at a value of 0.03345934
```

- `wordsum`, `colmil`, and `homosex` have the highest interaction strength in the linear model
- `zodiac`, `spend3`, and `sex` have the highest interaction strength in the elastic net model
- `degree`, `authoritarianism`, and `attend` have the highest interaction strength in the PCR model
- `social_connect`, `degree`, and `colhomo` have the highest interaction strength in the PLS model
- Terms like `polviews` and `degree` tend to have relatively high interaction strength. This is pretty intuitive: political views and level of education clearly affect many other aspects of a person's character. For example, `polviews` can affect people's choice on whether grass is legal and whether homosexuality is acceptable–and therefore affect `grass` and `homosex.degree`, on the other hand, influences `wordsum` and a general character of how people observe information (`news`, for example).
- PCR yields the term with the highest overall interaction strength across all models.
- The difference of interaction strength among terms is the largest in the elastic net model: this probably means that lasso and ridge methods reduce the influence of some of the features.

[ ]:
```

```