Homework 4: Moving Beyond Linearity

Wen Li Teng

February 16 2020

```
library(tidyverse)
library(caret)
library(margins)
library(splines)
library(ggplot2)
library(ggfortify)
library(robustHD)
library(glmnet)
library(pls)
library(iml)
library(h2o)

train <- read.csv("data/gss_train.csv")
test <- read.csv("data/gss_test.csv")

set.seed(1234)
```

1. POLYNOMIAL REGRESSION

Perform polynomial regression to predict `egalit_scale` as a function of `income06`

Use and plot 10-fold cross-validation

```
polynomial <- 1:10
poly_rsquared <- rep(0,10)
poly_RMSE <- rep(0,10)

train_control <- trainControl(method = "CV", number = 10)

for (i in 1:10) {
  poly_formula <- bquote(egalit_scale ~ poly(income06, .(i)))
  poly_mod <- train(as.formula(poly_formula),
                    data = train,
                    method = "lm",
                    trControl = train_control)
  poly_rsquared[i] <- poly_mod$results$Rsquared
  poly_RMSE[i] <- poly_mod$results$RMSE
}
```

```r
poly_table <- cbind(polynomial, poly_rsquared, poly_RMSE) %>%
  as.data.frame() %>%
  arrange(poly_RMSE)
poly_table
```

```
##    polynomial poly_rsquared poly_RMSE
## 1           9    0.06625946  9.326062
## 2           2    0.06323179  9.327615
## 3           3    0.06172276  9.330483
## 4           8    0.06119367  9.332842
## 5           6    0.05792979  9.340852
## 6           5    0.06439519  9.340872
## 7           7    0.06328038  9.341814
## 8           4    0.06304707  9.343810
## 9          10    0.06131416  9.349980
## 10          1    0.06256509  9.350900
```

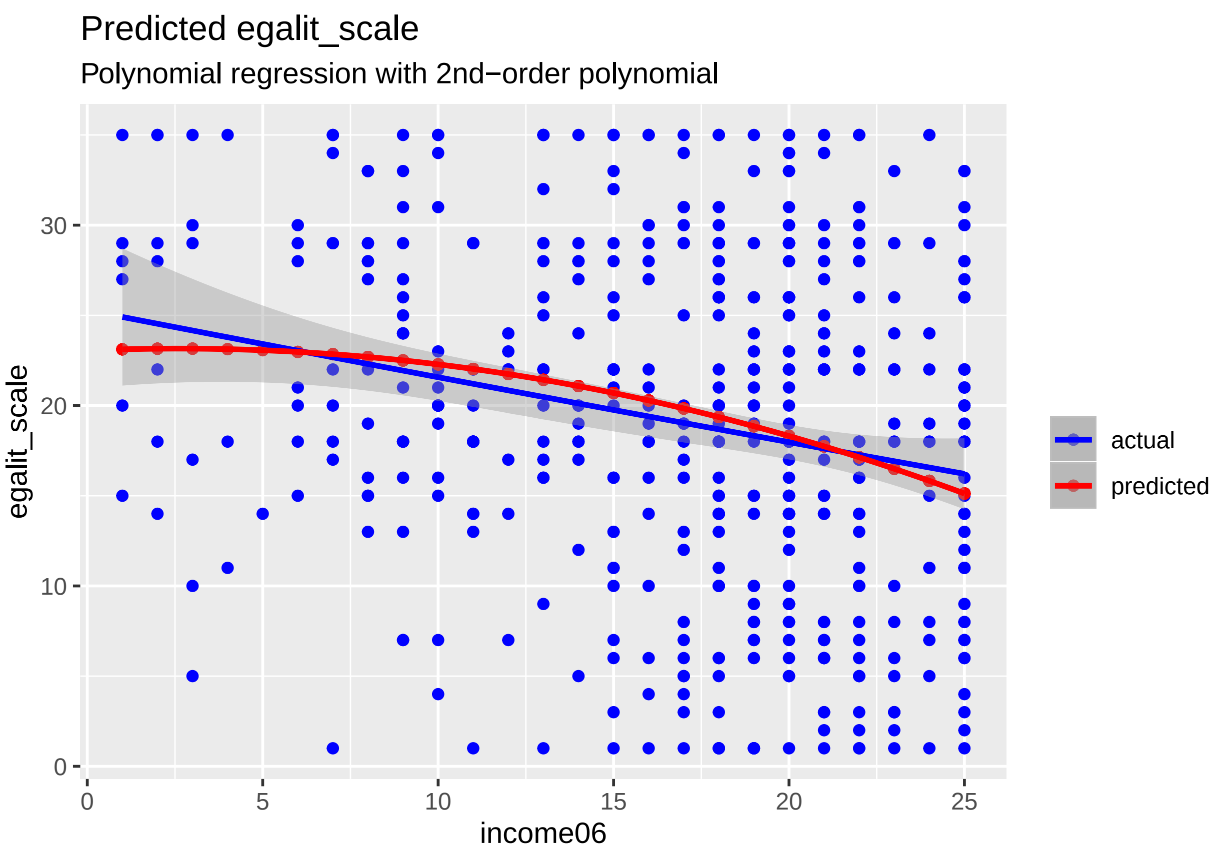
Select optimal degree for the polynomial based on MSE

Following the intuition of James et al. (2013), choose a polynomial that balances test error (accuracy) with simplicity (interpretability). The 9th-order polynomial is the most accurate, but is the 2nd-least interpretable. The 2nd-order polynomial is the 2nd-most accurate and is the 2nd-most interpretable. Therefore, choose the 2nd-order polynomial.

Plot polynomial fit to the data

```r
poly_mod <- lm(egalit_scale ~ poly(income06, 2), data = train)
poly_pred <- predict(poly_mod, newdata = test)
poly_df <- data.frame(income = test$income06,
                      actual = test$egalit_scale,
                      predicted = poly_pred)

ggplot(poly_df, aes(x = income)) +
  geom_point(aes(y = actual, color = "actual")) +
  geom_point(aes(y = predicted, color = "predicted")) +
  geom_smooth(method = "lm",
              formula = y ~ poly(x, 2),
              aes(y = actual, color = "actual")) +
  geom_smooth(data = poly_df,
              method = "lm",
              formula = y ~ poly(x, 2) ,
              aes(x = income, y = predicted,
              color = "predicted")) +
  scale_colour_manual("", values = c("actual" = "blue",
                                     "predicted" = "red")) +
  xlab("income06") + ylab("egalit_scale") +
  labs(title = "Predicted egalit_scale",
       subtitle = "Polynomial regression with 2nd-order polynomial")
```



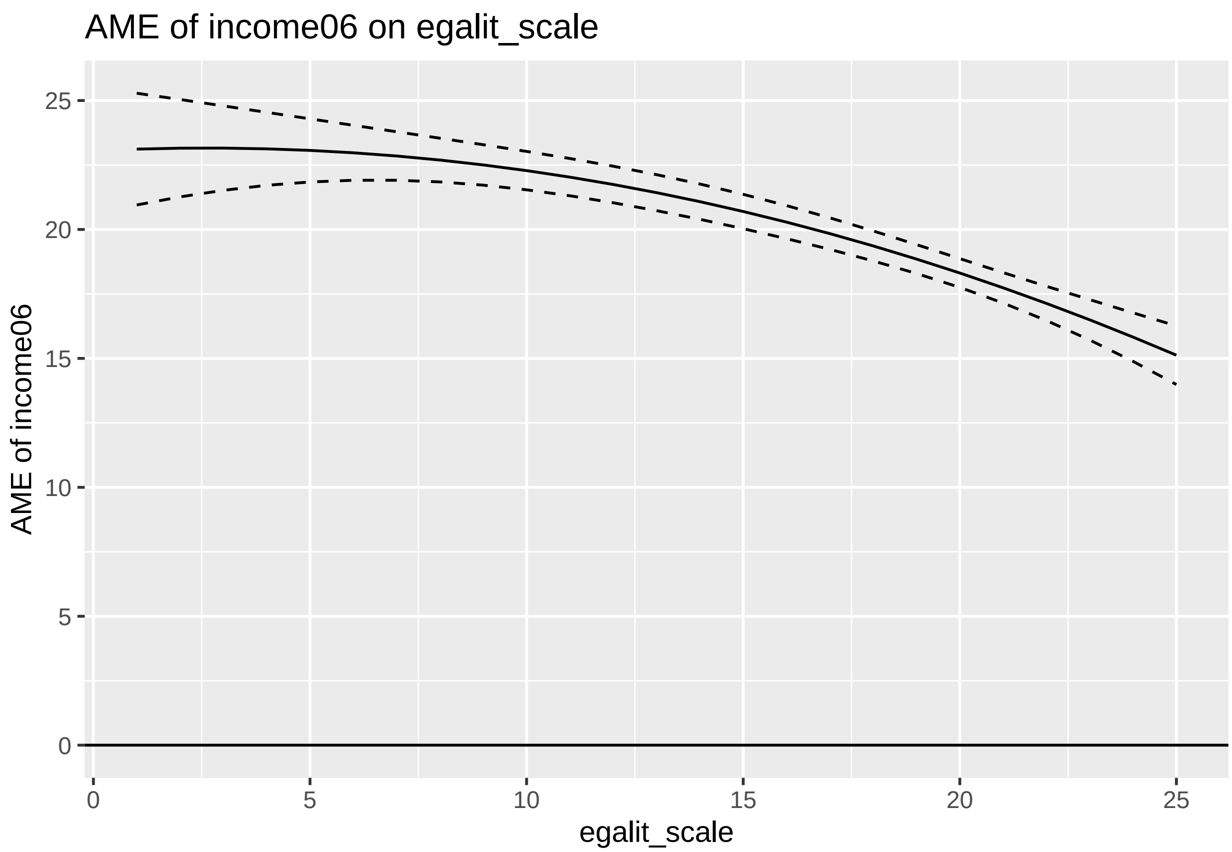
Graph the average marginal effect (AME) of `income06` across potential values

```
poly_dat <- cplot(poly_mod, "income06", draw = FALSE)
```

```
##    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
ggplot(poly_dat, aes(x = xvals)) +
  geom_line(aes(y = yvals)) +
  geom_line(aes(y = upper), linetype = 2) +
  geom_line(aes(y = lower), linetype = 2) +
  geom_hline(yintercept = 0) +
  ggtitle("AME of income06 on egalit_scale") +
  xlab("egalit_scale") + ylab("AME of income06")
```



Interpret the results

The polynomial regression model (with a 2nd-order polynomial) is of middling accuracy, judging from the R-squared values of the model and the fit of the model to the data. Nonetheless, the model can still be informative. In particular, the model reflects an inverse relationship between the `income06` and `egalit_scale`. The average marginal effects of `income06` decreases as `egalit_scale` increases. Put differently, income appears to have less of an effect on egalitarianism at higher levels of income.

2. STEP FUNCTION

Fit a step function to predict `egalit_scale` as a function of `income06`

Perform 10-fold cross-validation to choose optimal number of cuts

```
cut <- 1:10
step_rsquared <- rep(0,10)
step_RMSE <- rep(0,10)

for (i in 1:10) {
  step_formula <- bquote(egalit_scale ~ cut(income06, .(i+1)))
  step_mod <- train(as.formula(step_formula),
                    data = train,
                    method = "lm",
                    trControl = train_control)
  step_rsquared[i] <- step_mod$results$Rsquared
  step_RMSE[i] <- step_mod$results$RMSE
}

step_table <- cbind(cut, step_rsquared, step_RMSE) %>%
  as.data.frame() %>%
  arrange(step_RMSE)
step_table
```

```
##    cut step_rsquared step_RMSE
## 1    3    0.06475192  9.340726
## 2    6    0.05863575  9.352234
## 3    7    0.05603677  9.365507
## 4    4    0.05684573  9.369439
## 5    8    0.06325533  9.373954
## 6    5    0.05814558  9.391172
## 7   10    0.05184705  9.393802
## 8    9    0.05010242  9.402859
## 9    2    0.04826193  9.407046
## 10   1    0.04640243  9.471710
```

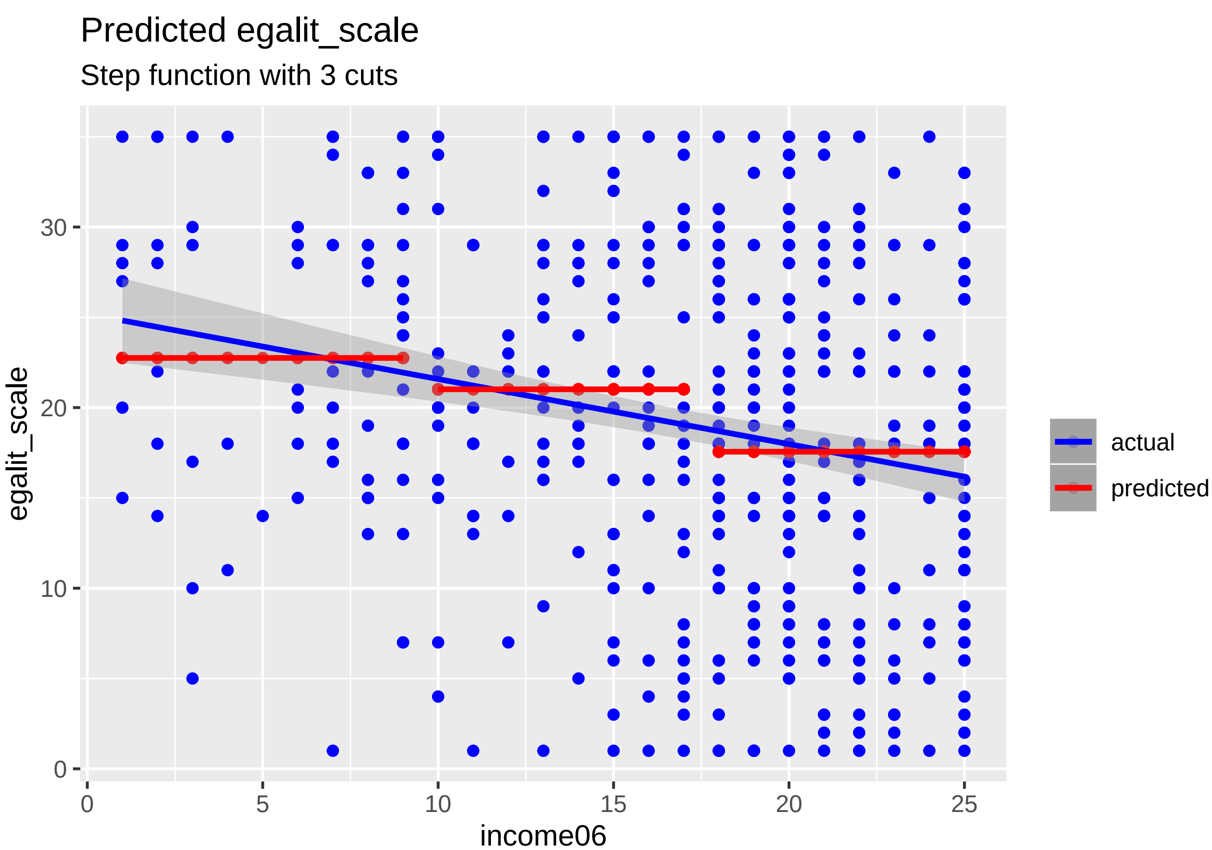
Select 3 as number of cuts, given that it has the lowest RMSE (accuracy) and is a relatively low number of cuts (interpretability).

Plot the fit

```
step_mod <- lm(egalit_scale ~ cut(income06, 3), data = train)
step_pred <- predict(step_mod, newdata = test)
step_df <- data.frame(income = test$income06,
                      actual = test$egalit_scale,
                      predicted = step_pred)
step_df1 <- step_df %>%
  filter(income <= 9)
```

```r
step_df2 <- step_df %>%
  filter(income > 9 & income <= 17)
step_df3 <- step_df %>%
  filter(income > 17)

ggplot(step_df, aes (x = income)) +
  geom_point(aes(y = actual, color = "actual")) +
  geom_point(aes(y = predicted, color = "predicted")) +
  geom_smooth(method = "lm", aes(y = actual, color = "actual")) +
  geom_smooth(data = step_df1, method = "lm", aes(y = predicted, color = "predicted")) +
  geom_smooth(data = step_df2, method = "lm", aes(y = predicted, color = "predicted")) +
  geom_smooth(data = step_df3, method = "lm", aes(y = predicted, color = "predicted")) +
  scale_colour_manual("", values = c("actual" = "blue",
                                    "predicted" = "red")) +
  xlab("income06") + ylab("egalit_scale") +
  labs(title = "Predicted egalit_scale",
       subtitle = "Step function with 3 cuts")
```



Interpret the results

Like the polynomial regression model above, the step function (with 3 cuts) is of middling accuracy, judging from the R-squared values of the model and the fit of the model to the data.

3. NATURAL REGRESSION SPLINE

Fit a natural regression spline to predict `egalit_scale` as a function of `income06`

Use 10-fold cross-validation to select optimal number of degrees of freedom

```
degree <- 1:10
natreg_rsquared <- rep(0,10)
natreg_RMSE <- rep(0,10)

for (i in 1:10) {
  natreg_formula <- bquote(egalit_scale ~ ns(income06, df = .(i)))
  natreg_mod <- train(as.formula(natreg_formula),
                      data = train,
                      method = "lm",
                      trControl = train_control)
  natreg_rsquared[i] <- natreg_mod$results$Rsquared
  natreg_RMSE[i] <- natreg_mod$results$RMSE
}

natreg_table <- cbind(degree, natreg_rsquared, natreg_RMSE) %>%
  as.data.frame() %>%
  arrange(natreg_RMSE)
natreg_table
```

```
##    degree natreg_rsquared natreg_RMSE
## 1       3      0.06335959    9.325581
## 2       2      0.06784346    9.337192
## 3      10      0.06219866    9.337570
## 4       7      0.06458164    9.343013
## 5       1      0.06037112    9.343996
## 6       9      0.06189211    9.350200
## 7       8      0.05890278    9.351720
## 8       5      0.05830273    9.352151
## 9       6      0.06068609    9.353680
## 10      4      0.05981056    9.353758
```

Select 3 as number of degrees of freedom, given that it has the lowest RMSE (accuracy) and is a relatively low number of degrees (interpretability).

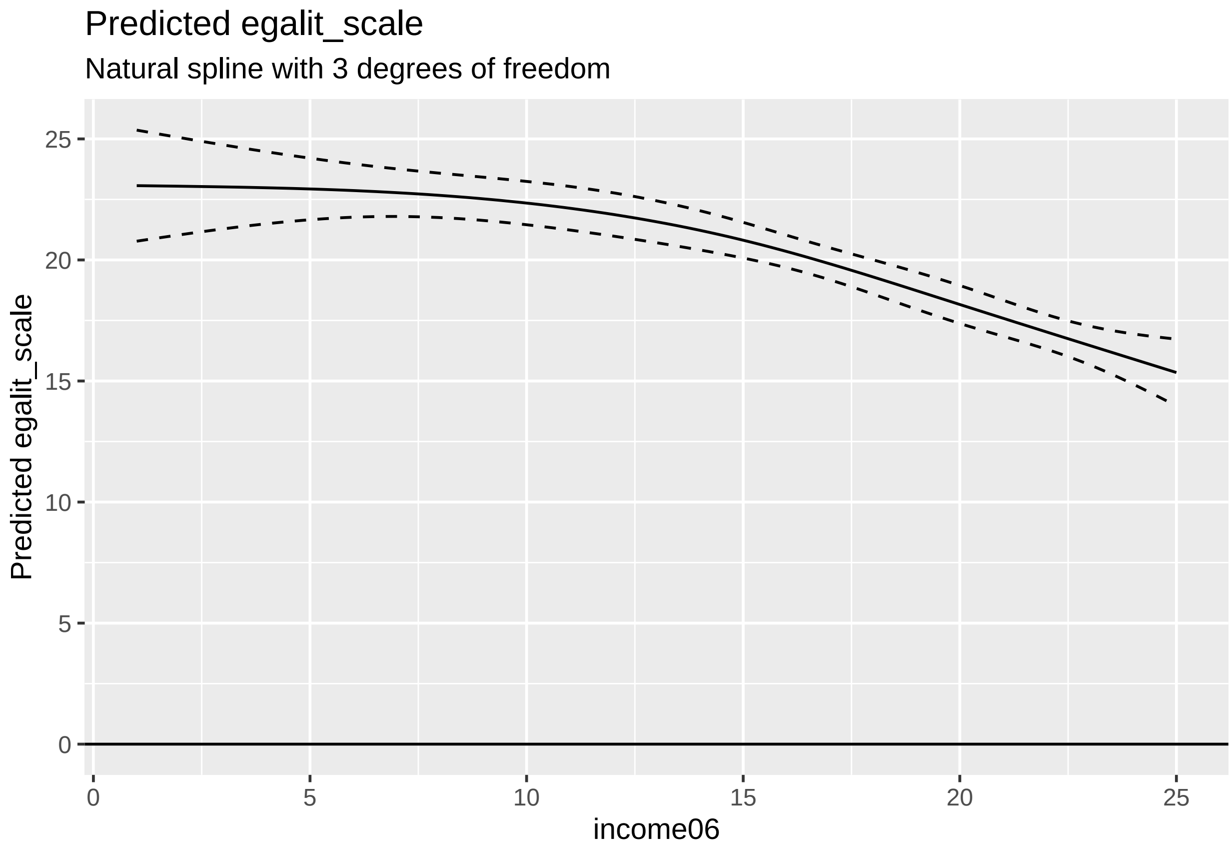
```
natreg_mod <- lm(egalit_scale ~ ns(income06, df = 3), data = train)
natreg_pred <- predict(step_mod, newdata = test)

natreg_df <- data.frame(income = test$income06,
                        actual = test$egalit_scale,
                        predicted = natreg_pred)
```

Present the results of the optimal model

```r
cplot(natreg_mod, "income06", what = "prediction", n = 101, draw = FALSE) %>%
  ggplot(aes(x = xvals)) +
  geom_line(aes(y = yvals)) +
  geom_line(aes(y = upper), linetype = 2) +
  geom_line(aes(y = lower), linetype = 2) +
  geom_hline(yintercept = 0, linetype = 1) +
  labs(title = "Predicted egalit_scale",
       subtitle = "Natural spline with 3 degrees of freedom",
       x = "income06",
       y = "Predicted egalit_scale")
```

```
##    xvals    yvals    upper    lower
## 1   1.00 23.06899 25.36250 20.77548
## 2   1.24 23.06347 25.28680 20.84015
## 3   1.48 23.05790 25.21163 20.90417
## 4   1.72 23.05221 25.13704 20.96738
## 5   1.96 23.04635 25.06307 21.02962
## 6   2.20 23.04025 24.98976 21.09073
## 7   2.44 23.03386 24.91716 21.15055
## 8   2.68 23.02711 24.84533 21.20890
## 9   2.92 23.01996 24.77432 21.26561
## 10  3.16 23.01235 24.70420 21.32049
## 11  3.40 23.00420 24.63502 21.37338
## 12  3.64 22.99547 24.56687 21.42408
## 13  3.88 22.98610 24.49981 21.47240
## 14  4.12 22.97603 24.43392 21.51815
## 15  4.36 22.96520 24.36929 21.56112
## 16  4.60 22.95356 24.30599 21.60112
## 17  4.84 22.94103 24.24411 21.63795
## 18  5.08 22.92757 24.18374 21.67140
## 19  5.32 22.91312 24.12495 21.70129
## 20  5.56 22.89762 24.06782 21.72742
```



The natural spline model (with 3 degrees of freedom) shows that there is an inverse relationship between `income06` and `egalit_scale`. The higher an individual's income, the less egalitarian they are likely to be. As such, the natural spline model supports the general findings from the earlier polynomial regression model.

4(a). LINEAR REGRESSION

Perform appropriate data pre-processing

Standardize numerical features

```
test <- test %>%
  mutate_if(is.integer, standardize)

train <- train %>%
  mutate_if(is.integer, standardize)
```

Estimate model using all available predictors

Use 10-fold cross-validation to estimate model's performance using MSE

```r
train_control <- trainControl(method = "CV", number = 10)
linear_mod <- train(egalit_scale ~ .,
                    data = train,
                    method = "lm",
                    trControl = train_control)
linear_pred <- linear_mod %>% predict(test)
linear_mse <- (RMSE(linear_pred, test$egalit_scale))^2
linear_mse
```

```
## [1] 0.6954183
```

4(b). ELASTIC NET REGRESSION

Perform appropriate hyperparameter tuning

Estimate model using all available predictors

Use 10-fold cross-validation to estimate model's performance using MSE

```r
enet_mod <- train(egalit_scale ~ .,
                  data = train,
                  method = "glmnet",
                  trControl = train_control,
                  tuneGrid = expand.grid(alpha = seq(0, 1, 0.1),
                                         lambda = seq(0.001, 0.1, 0.001)))

enet_pred <- enet_mod %>% predict(test)
enet_mse <- (RMSE(enet_pred, test$egalit_scale))^2
enet_mse
```

```
## [1] 0.6775607
```

4(c). PRINCIPAL COMPONENT REGRESSION

Perform appropriate hyperparameter tuning

Estimate model using all available predictors

```r
pcr_mod <- train(egalit_scale ~ .,
                 data = train,
                 method = "pcr",
                 trControl = train_control,
                 tuneLength = 10)
```

Use 10-fold cross-validation to estimate model's performance using MSE

```
pcr_pred <- pcr_mod %>% predict(test)
pcr_mse <- (RMSE(pcr_pred, test$egalit_scale))^2
pcr_mse
```

```
## [1] 0.8520385
```

4(d). PARTIAL LEAST SQUARES REGRESSION

Perform hyperparameter tuning

Estimate model using all available predictors

```
pls_mod <- train(egalit_scale ~ .,
                 data = train,
                 method = "pls",
                 trControl = train_control,
                 tuneLength = 10)
```

Use 10-fold cross-validation to estimate model's performance using MSE

```
pls_pred <- pls_mod %>% predict(test)
pls_mse <- (RMSE(pls_pred, test$egalit_scale))^2
pls_mse
```

```
## [1] 0.6866766
```

5. FEATURE INTERACTION PLOTS

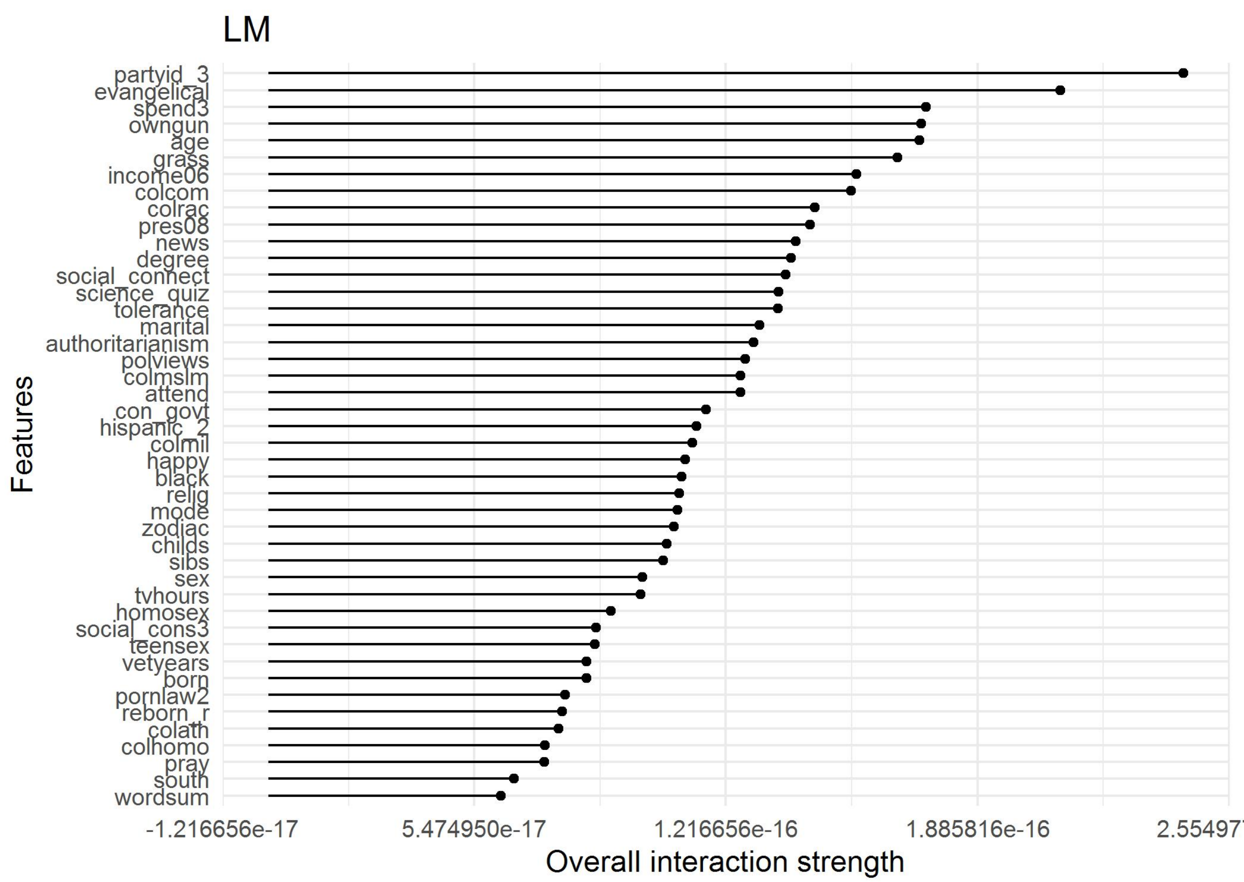
Linear Regression

Plot

```
predictor_lm <- Predictor$new(
  model = linear_mod,
  data = features,
  y = response,
  predict.fun = pred,
  class = "classification"
)
```

```r
interact_lm <- Interaction$new(predictor_lm)
```

```r
plot(interact_lm) +
  ggtitle("LM") +
  theme_minimal(base_size = 12)
```



Discussion

In this linear model, `party_id3`, `evangelical`, and `spend3` are the three features with the highest interaction strength. `partyid3` is by far the strongest, followed by `evangelical`. `spend3` is not much stronger than the variables after it.

Elastic Net Regression

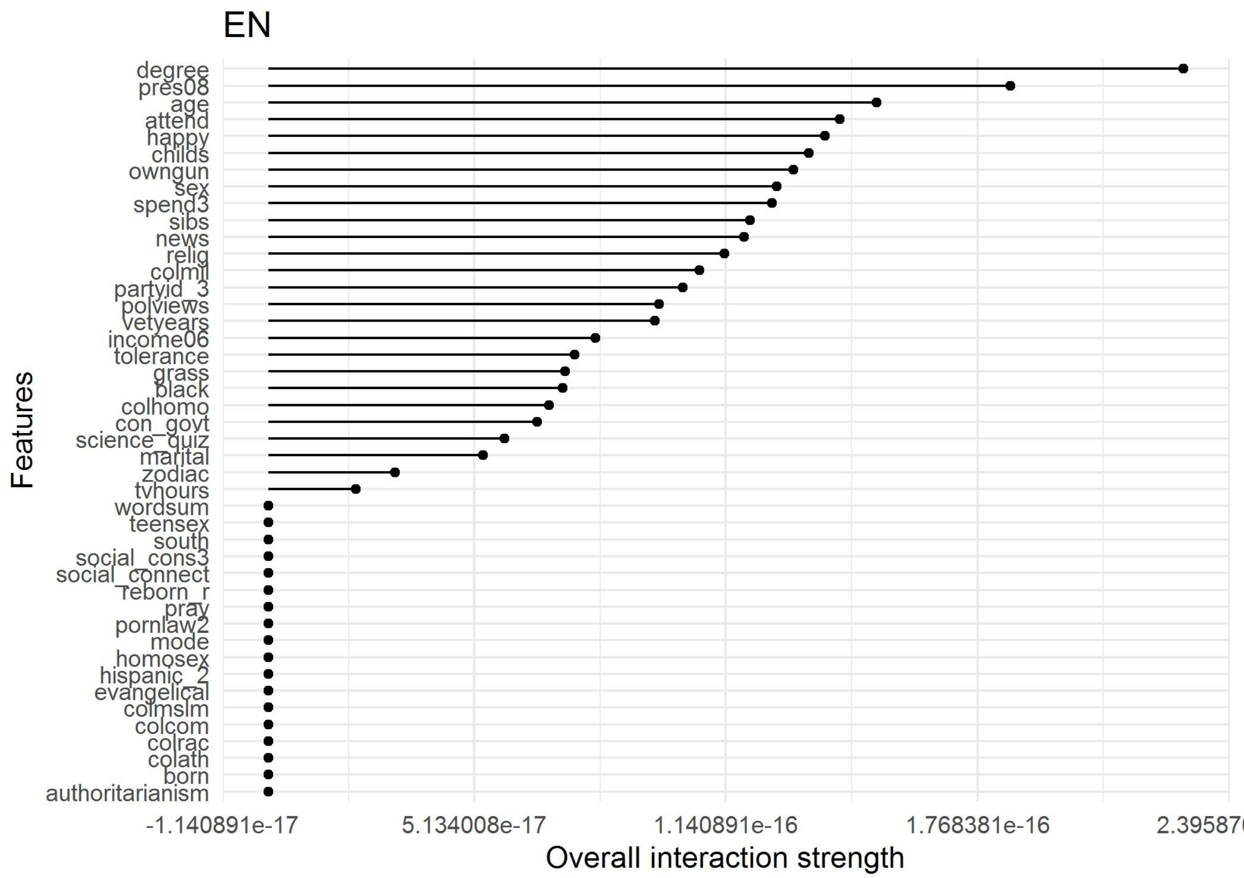
Plot

```r
predictor_enet <- Predictor$new(
  model = enet_mod,
  data = features,
  y = response,
  predict.fun = pred,
```

```
  class = "classification"
)
```

```
interact_enet <- Interaction$new(predictor_enet)
```

```
plot(interact_enet) +
  ggtitle("EN") +
  theme_minimal(base_size = 12)
```



Discussion In this elastic net model, `degree`, `pres08`, and `age` are the three features with the highest interaction strength. `degree` is by far the strongest, followed by `pres08`. `age08` is only marginally stronger than the variables after it. Many variables are of extremely low interaction strength, from `wordsum` to `authoritarianism`.

Principal Component Regression

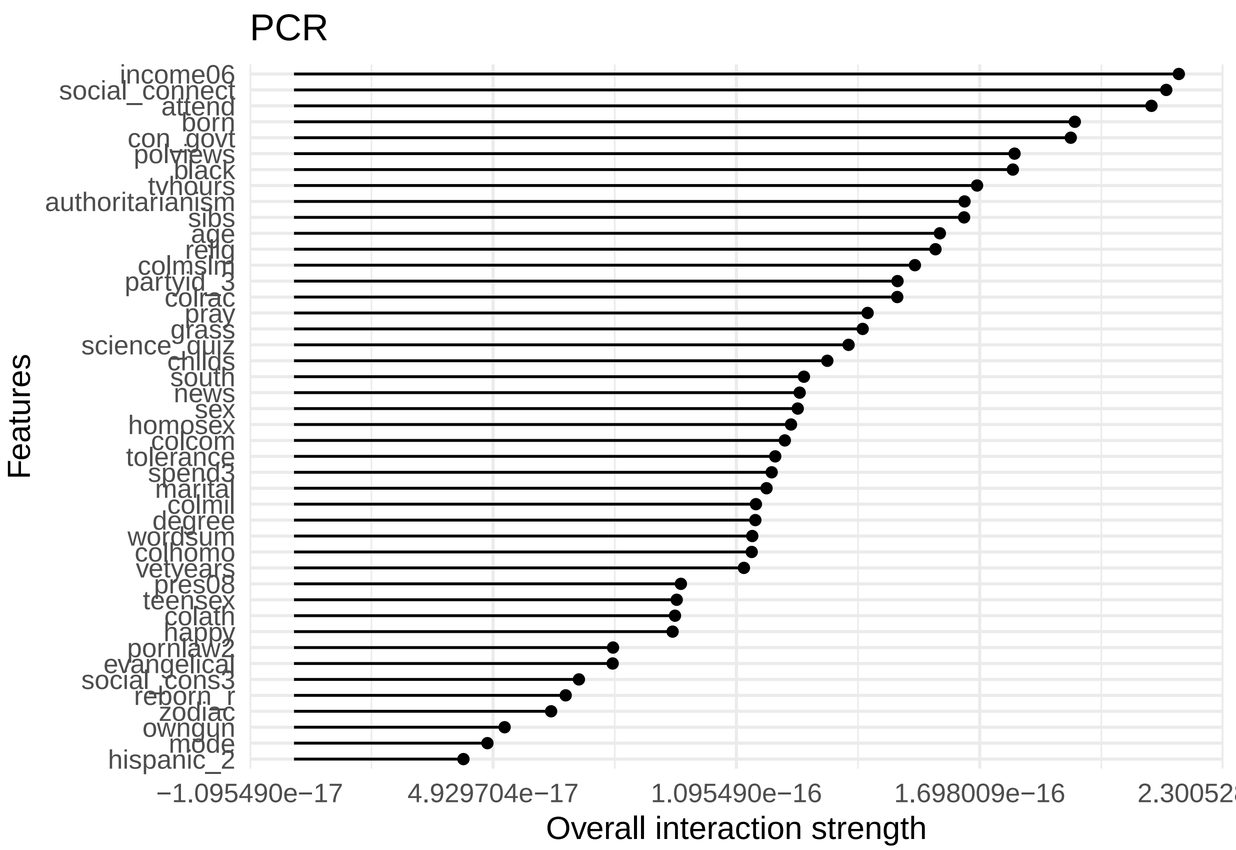
Plot

```
predictor_pcr <- Predictor$new(
  model = pcr_mod,
  data = features,
  y = response,
  predict.fun = pred,
```

```r
  class = "classification"
)
```

```r
interact_pcr <- Interaction$new(predictor_pcr)
```

```r
plot(interact_pcr) +
  ggtitle("PCR") +
  theme_minimal(base_size = 12)
```



Discussion

In this PCR model, `income06`, `social_connect`, and `attend` are the three features with the highest interaction strength. The three features are clustered together, with interaction strengths that are significantly higher than the next two features (`born` and `con_govt`).

Partial Least Squares Regression

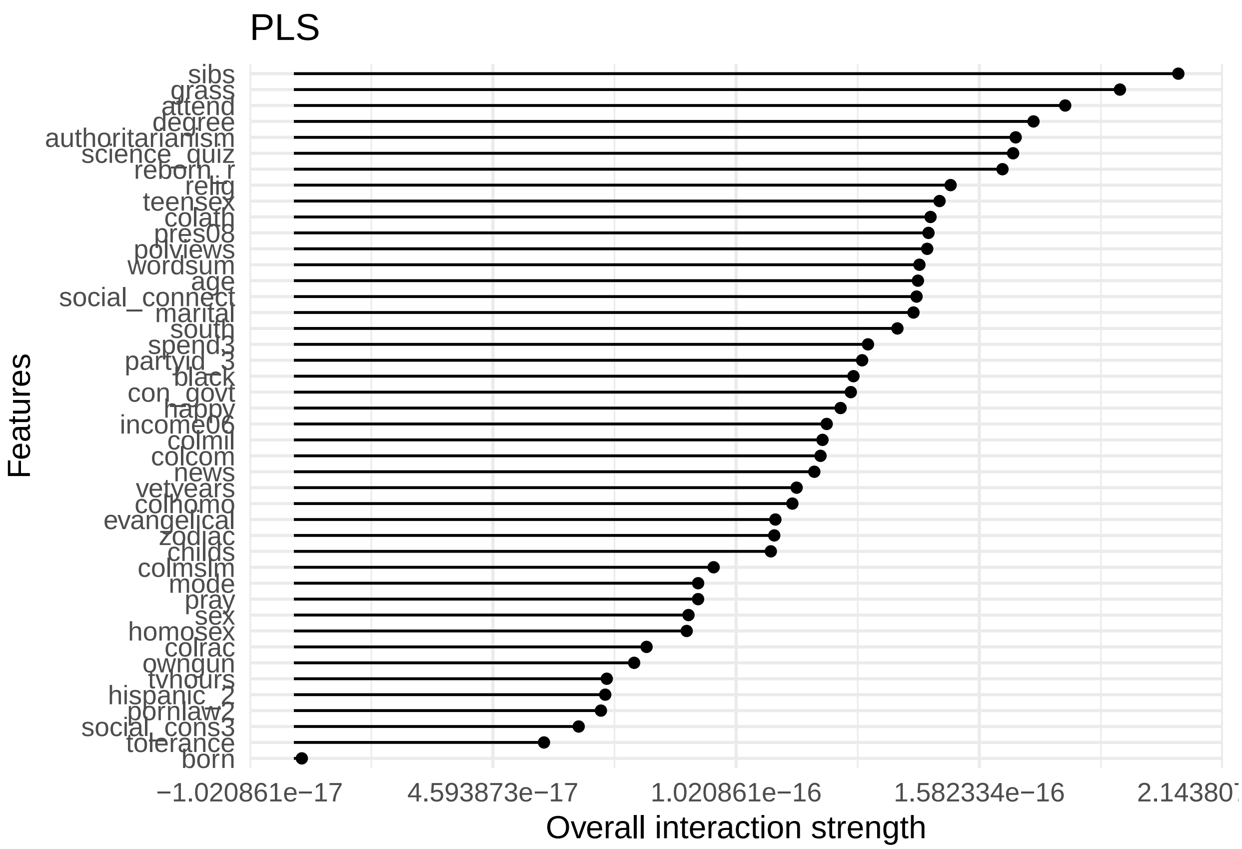
Plot

```r
predictor_pls <- Predictor$new(
  model = pls_mod,
  data = features,
```

```r
  y = response,
  predict.fun = pred,
  class = "classification"
)
```

```r
interact_pls <- Interaction$new(predictor_pls)
```

```r
plot(interact_pls) +
  ggtitle("PLS") +
  theme_minimal(base_size = 12)
```



Discussion

In this PLS model, `sibs`, `grass`, and `attend` are the three features with the highest interaction strength. `sibs` is by far the strongest. `grass` and `attend` follow, decreasing in strength by about the same quantities.

Comparison

Observe that the top three features of each model are largely different:
* LM: `partyid_3`, `evangelical`, `spend3`
* EN: `degree`, `pres08`, `age`
* PCR: `income06`, `social_connect`, `attend`
* PLS: `sibs`, `grass`, `attend`
The only repeated feature is `attend`.

Observe also that the clustering of feature interaction strengths differs between models:
* LM: 1 feature that is clearly strongest, 1 feature that is clearly the next-strongest, followed by many other features of noticeable strength
* EN: 1 feature that is clearly strongest, 1 feature that is clearly the next-strongest, followed by several other features of noticeable strength, and several other features of negligible strength
* PCR: 3 features that are clearly strongest, followed by small clusters of features of decreasing strength
* PLS: 1 feature that is clearly strongest, 1 feature that is clearly the next-strongest, 1 feature that is the next-strongest after, and large clusters of features of decreasing strength, ending with 1 feature of negligible strength
The differences between the models may suggest that some models are better suited to the GSS data than others, or even that the models examined may not capture the actual features that weigh the most heavily on the response. In either cases, different models carry different assumptions about the data, which influence how the models identify the most important features.