# Appendix D
# Replication scripts: Models fitted in main paper

## Introduction

In this Appendix, we present scripts for replicating the results from the main paper. In the Method section, packages used are listed and data preparation is shown. In the Results section, we show how each of the models in the main document were fit, and how the figures and tables can be replicated. In sections Appendix B and C, we show how results, tables and figures from the appendices can be replicated. Finally, the last section provides the references.

## Method

All analyses were performed in **R** (version 4.3.0, R Core Team, 2021). We fitted (penalized) logistic regression models using **R** package **glmnet** (version 4.1.6, Friedman et al., 2010); generalized additive models (GAMs) with smoothing splines using package **mgcv** (version 1.8.41, Wood, 2017); conditional inference trees using package **partykit** (version 1.2.16, Hothorn et al., 2006); gradient boosted tree ensembles using package **gbm** (version 2.1.8.1, Greenwell et al., 2020); random forests using package **ranger** (version 0.14.1, Wright & Ziegler, 2017); prediction rule ensembles using package **pre** (version 1.0.7, Fokkema, 2020); $k$ nearest neighbours using package **class** (version 7.3.21, Venables & Ripley, 2002).

Appendix E shows our code and results for tuning the parameter settings using cross validation. All parameter settings employed here are based on those results, except for GAMs. We did not tune the parameters of the GAMs with smoothing splines, because we expected the defaults to work well out of the box. We simply employed the defaults of thin-plate regression splines and generalized cross validation method to fit the smoothing splines. This provides built-in regularization of the wigglyness, without the need for choosing an optimal value for penalty or tuning parameters. For smaller samples, restricted maximum likelihood (REML) would be preferred. For the analyses using the items as predictors, we set the number of basis functions of the thin-plate regression splines to four, instead of the default of eight, because the item responses have only five possible values; increasing the number of basis functions would yield an unidentified model. Furthermore, because of the larger number of predictors, and the benefit of penalization observed in the logistic regression, we also applied a penalty to the linear part of each smoothing spline function, so that some items can be completely eliminated from the model.

## Data preparation

```
data <- read.delim("data.csv", header = TRUE)

## Items should be scored 1-5, 0s are missings
data[ , 1:48][sapply(data[ , 1:48], function(x) x == 0)] <- NA
data <- data[complete.cases(data[ , 1:48]), ]

## Select only university students
data <- data[data$education >= 3, ]
# table(data$major)
psych_ids <- rowSums(sapply(c("psych", "psyhcology", "psycotherapy", "couns",
                              "behavior", "behaviour", "neuro"),
                            function(x) grepl(x, data$major, ignore.case = TRUE)))
anim_ids <- grepl("anim", data$major, ignore.case = TRUE) ## exclude animal psych
data$major <- factor(ifelse(psych_ids > 0, "psychology", "other"))
data$major[anim_ids > 0 & psych_ids > 0] <- "other"

set.seed(42)
test_ids <- sample(1:nrow(data), ceiling(nrow(data)/4))
train_ids <- which(!1:nrow(data) %in% test_ids)
train_y <- as.numeric(data$major)[train_ids] - 1
test_y <- as.numeric(data$major)[test_ids] - 1

data$Real <- rowSums(data[ , paste0("R", 1:8)])
data$Inve <- rowSums(data[ , paste0("I", 1:8)])
data$Arti <- rowSums(data[ , paste0("A", 1:8)])
data$Soci <- rowSums(data[ , paste0("S", 1:8)])
data$Ente <- rowSums(data[ , paste0("E", 1:8)])
data$Conv <- rowSums(data[ , paste0("C", 1:8)])
```

Sample size:

```
nrow(data)
```

```
## [1] 55593
```

Proportions of observations choosing psychology (not) as a major:

```
prop.table(table(data$major)) ## total dataset
```

```
##
##      other psychology
##  0.8057849  0.1942151
```

```r
prop.table(table(data$major[train_ids])) ## train dataset
```

```
##
##      other psychology
##  0.8053677  0.1946323
```

```r
prop.table(table(data$major[test_ids])) ## test dataset
```

```
##
##      other psychology
##  0.8070365  0.1929635
```

Which yields the following variances:

```r
var(as.numeric(data$major[train_ids]))
```

```
## [1] 0.1567543
```

```r
var(as.numeric(data$major[test_ids]))
```

```
## [1] 0.1557398
```

# Results

```r
varnames_i <- paste0(rep(c("R", "I", "A", "S", "E", "C"), each = 8), 1:8)
varnames_s <- c("R", "I", "A", "S", "E", "C")
```

## (Penalized) Logistic Regression

```r
glmod_s <- glm(major ~ Real + Inve + Arti + Soci + Ente + Conv,
               data = data[train_ids , ], family = "binomial")
#summary(glmod_s)
glm_preds_train_s <- predict(glmod_s, newdata = data[train_ids, ], type = "response")
glm_preds_test_s <- predict(glmod_s, newdata = data[test_ids, ], type = "response")
```

```r
library("glmnet")
X <- as.matrix(data[train_ids, varnames_i])
set.seed(42)
glmod_i <- glmnet(X, train_y, family = "binomial", alpha = 1, lambda = 0.0003568404)
```

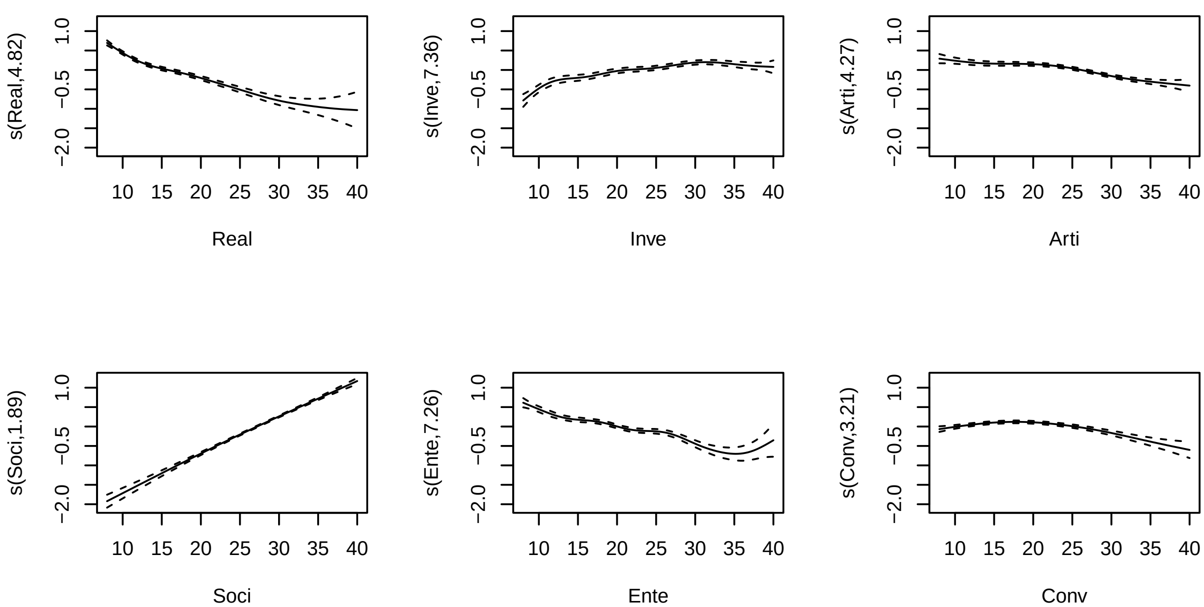
```r
glm_preds_train_i <- predict(glmod_i, newx = X, type = "response")
glm_preds_test_i <- predict(glmod_i, newx = as.matrix(data[test_ids, varnames_i]),
                            type = "response")
```

### Generalized Additive Model

```r
library("mgcv")
gamod_s <- gam(major ~ s(Real) + s(Inve) + s(Arti) + s(Soci) + s(Ente) + s(Conv),
               data = data[train_ids , ], family = "binomial")
```

#### Figure 2

```r
par(mfrow = c(2, 3))
plot(gamod_s)
```




```r
gam_preds_train_s <- predict(gamod_s, newdata = data[train_ids, ], type = "response")
gam_preds_test_s <- predict(gamod_s, newdata = data[test_ids, ], type = "response")
```

```r
## Construct item-level formula and fit GAM
gam_form <- formula(paste0("major ~", paste0("s(", varnames_i, ", k=4)",
                                              collapse = " + ")))
```

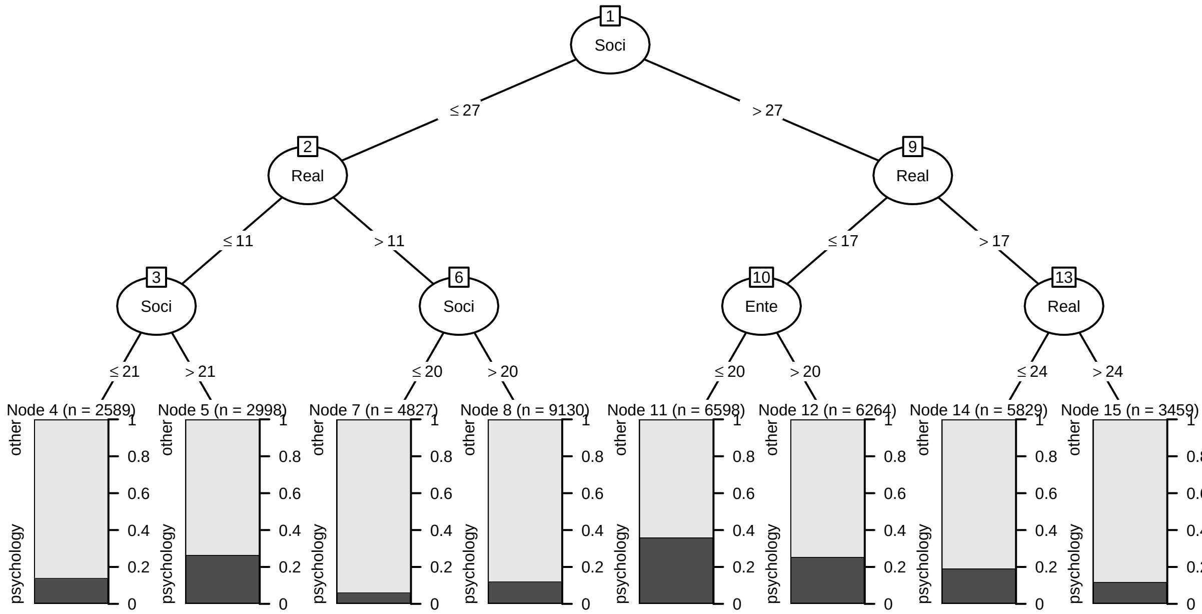
```
gamod_i <- gam(gam_form, data = data[train_ids , ], family = "binomial",
               select = TRUE)
```

```
gam_preds_train_i <- predict(gamod_i, newdata = data[train_ids, ], type = "response")
gam_preds_test_i <- predict(gamod_i, newdata = data[test_ids, ], type = "response")
```

## Decision Tree

### Figure 3

```
library("partykit")
ct_s <- ctree(major ~ Real + Inve + Arti + Soci + Ente + Conv, data = data[train_ids , ],
              maxdepth = 7)
ct3 <- ctree(major ~ Real + Inve + Arti + Soci + Ente + Conv, data = data[train_ids , ],
             maxdepth = 3)
plot(ct3, gp = gpar(cex = .5), ip_args = list(pval = FALSE))
```




```
ct_preds_train_s <- predict(ct_s, type = "prob")[ , "psychology"]
ct_preds_test_s <- predict(ct_s, newdata = data[test_ids , ], type = "prob")[ , "psychology"]
```

```
ct_form <- formula(paste("major ~", paste(varnames_i, collapse = "+")))
ct <- ctree(ct_form, data = data[train_ids , ], maxdepth = 9)
ct_preds_train_i <- predict(ct, type = "prob")[ , "psychology"]
ct_preds_test_i <- predict(ct, newdata = data[test_ids , ], type = "prob")[ , "psychology"]
```

## Gradient Boosted Tree Ensemble

```
library("gbm")
set.seed(42)
gb_s <- gbm(I(as.numeric(major)-1) ~ Real + Inve + Arti + Soci + Ente + Conv,
            n.trees = 1100, interaction.depth = 3L, shrinkage = 0.01,
            data = data[train_ids , ])
```

```
gb_preds_train_s <- predict(gb_s, newdata = data[train_ids, ], type = "response")
gb_preds_test_s <- predict(gb_s, newdata = data[test_ids, ], type = "response")
```

```
library("gbm")
set.seed(42)
gbm_form <- formula(paste("I(as.numeric(major)-1) ~ ",
                          paste(paste0(rep(c("R", "I", "A", "S", "E", "C"),
                                           each = 8), 1:8), collapse = "+")))
gb_i <- gbm(gbm_form, n.trees = 3500, interaction.depth = 5L, shrinkage = 0.01,
            data = data[train_ids , ])
sum_i <- summary(gb_i, plotit = FALSE, method = permutation.test.gbm)
```

```
gb_preds_train_i <- predict(gb_i, newdata = data[train_ids, ], type = "response")
gb_preds_test_i <- predict(gb_i, newdata = data[test_ids, ], type = "response")
```

## Random Forest

```
library("ranger")
set.seed(42)
rf_s <- ranger(major ~ Real + Inve + Arti + Soci + Ente + Conv, data = data[train_ids , ],
                   probability = TRUE, mtry = 3L, min.node.size = 500,
             importance = "permutation")
```

```
rf_preds_train_s <- predict(rf_s, data = data[train_ids, ])$predictions[ , "psychology"]
rf_preds_test_s <- predict(rf_s, data = data[test_ids, ])$predictions[ , "psychology"]
```

```
set.seed(42)
varnames <- paste0(rep(c("R", "I", "A", "S", "E", "C"), each = 8), 1:8)
rf_form <- formula(paste("major ~", paste(varnames, collapse = "+")))
rf_i <- ranger(rf_form, data = data[train_ids , ], probability = TRUE,
             mtry = 10L, min.node.size = 500, importance = "permutation")
```

```r
rf_preds_train_i <- predict(rf_i, data = data[train_ids, ])$predictions[ , "psychology"]
rf_preds_test_i <- predict(rf_i, data = data[test_ids, ])$predictions[ , "psychology"]
```

### Prediction Rule Ensemble

```r
library("pre")
set.seed(42)
pr_s <- pre(major ~ Real + Inve + Arti + Soci + Ente + Conv, data = data[train_ids , ],
            learnrate = .05)
```

#### Table 1

```r
pr_preds_train_s <- predict(pr_s, type = "response")
pr_preds_test_s <- predict(pr_s, newdata = data[test_ids , ], type = "response")
imps <- pre::importance(pr_s, plot=FALSE)
varimps_pre <- imps$varimps
imps <- imps$baseimps[1:6, c("description", "coefficient")]
colnames(imps) <- c("Description", "Coefficient")
imps$Coefficient <- round(imps$Coefficient, digits = 3)
knitr::kable(imps, row.names = FALSE, align = c("l", "c"))
```

| Description | Coefficient |
|:---|:---:|
| Soci > 27 & Ente <= 31 & Conv <= 30 | 0.182 |
| Soci > 23 & Ente <= 29 & Real <= 24 | 0.181 |
| Real > 10 & Soci <= 35 | -0.175 |
| Real <= 22 & Soci > 19 & Inve > 18 | 0.138 |
| Inve > 10 & Real <= 13 | 0.120 |
| Conv <= 23 & Arti <= 29 & Soci > 21 | 0.112 |

```r
pr_form <- formula(paste("major ~", paste(varnames_i, collapse = "+")))
set.seed(42)
pr_i <- pre(pr_form, data = data[train_ids , ], learnrate = .05,
            verbose = TRUE, family = "binomial")
```

```r
pr_preds_train_i <- predict(pr_i, type = "response")
pr_preds_test_i <- predict(pr_i, newdata = data[test_ids , ], type = "response")
```

## *k* Nearest Neighbours

```r
library("class")
ss_names <- c("Real", "Inve", "Arti", "Soci", "Ente", "Conv")
## Model for training predictions
knn_mod <- knn(train = data[train_ids , ss_names],
               test = data[train_ids , ss_names],
               cl = as.factor(data[train_ids, "major"]),
               k = 300, use.all = TRUE, prob = TRUE)
## Need to obtain predicted probability for second class
knn_preds_train_s <- ifelse(
  knn_mod == "psychology", attr(knn_mod, "prob"), 1 - attr(knn_mod, "prob"))
```

```r
## Model for testing predictions
knn_mod <- knn(train = data[train_ids , ss_names],
               test = data[test_ids , ss_names],
               cl = as.factor(data[train_ids, "major"]),
               k = 300, use.all = TRUE, prob = TRUE)
knn_preds_test_s <- ifelse(
  knn_mod == "psychology", attr(knn_mod, "prob"), 1 - attr(knn_mod, "prob"))
```

```r
## Model for training predictions
knn_mod <- knn(train = data[train_ids , varnames_i],
               test = data[train_ids , varnames_i],
               cl = as.factor(data[train_ids, "major"]),
               k = 100, use.all = TRUE, prob = TRUE)
## Need to obtain predicted probability for second class
knn_preds_train_i <- ifelse(
  knn_mod == "psychology", attr(knn_mod, "prob"), 1 - attr(knn_mod, "prob"))
```

```r
## Model for testing predictions
knn_mod <- knn(train = data[train_ids , varnames_i],
               test = data[test_ids , varnames_i],
               cl = as.factor(data[train_ids, "major"]),
               k = 100, use.all = TRUE, prob = TRUE)
knn_preds_test_i <- ifelse(
  knn_mod == "psychology", attr(knn_mod, "prob"), 1 - attr(knn_mod, "prob"))
```

## Model comparisons

### Figure 1

```
par(mfrow = c(2, 3))
par(mar = c(1, 4, 2, 1), mgp = c(1.5, .5, 0), tck = -0.05)
library("colorspace")

## Logistic regression
plot(coef(glmod_s)[-1], xaxt = "n", ylab = "Estimated coefficient",
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = "Logistic Regression", cex.main = .7)
text(coef(glmod_s)[-1], labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)
abline(0, 0, col = "grey")

## Generalized additive model
sum <- summary(gamod_s)
plot(sqrt(sum$chi.sq), xaxt = "n", ylab = expression(sqrt(chi^2)),
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = "Generalized Additive Model", cex.main = .7)
text(sqrt(sum$chi.sq), labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)

## Conditional inference tree
ct6 <- cforest(major ~ Real + Inve + Arti + Soci + Ente + Conv,
               data = data[train_ids , ], ntree = 1L, mtry = 6,
               perturb = list(replace = FALSE, fraction = 1L),
               control = ctree_control(maxdepth = 6))
imps <- varimp(gettree(ct6), risk = "loglik")
imps <- imps[c("Real", "Inve", "Arti", "Soci", "Ente", "Conv")]
plot(sqrt(imps), xaxt = "n", ylab = expression(sqrt(Importance)),
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = "Tree", cex.main = .7)
text(sqrt(imps), labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)

## Gradient boosted ensemble
sum <- summary(gb_s, plotit = FALSE, method = permutation.test.gbm)
imps <- sum$rel.inf
names(imps) <- sum$var
imps <- imps[c("Real", "Inve", "Arti", "Soci", "Ente", "Conv")]
plot(sqrt(imps), xaxt = "n", ylab = expression(sqrt(Importance)),
```

```
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = "Gradient Boosted Ensemble", cex.main = .7)
text(sqrt(imps), labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)

## Random forest
library("ranger")
load(file = "RF_subscales.Rda")
imps <- ranger::importance(rf_s)
plot(sqrt(imps), xaxt = "n", ylab = expression(sqrt(Importance)),
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = "Random Forest", cex.main = .7)
text(sqrt(imps), labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)

## Prediction rule ensemble
imps <- varimps_pre$imp
names(imps) <- varimps_pre$varname
imps <- imps[c("Real", "Inve", "Arti", "Soci", "Ente", "Conv")]
plot(imps, xaxt = "n", ylab = "Importance",
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     cex.main = .7, main = "Prediction Rule Ensemble")
text(imps, labels = varnames_s, cex = 1,
     col = rep(qualitative_hcl(6)), font = 2)
```

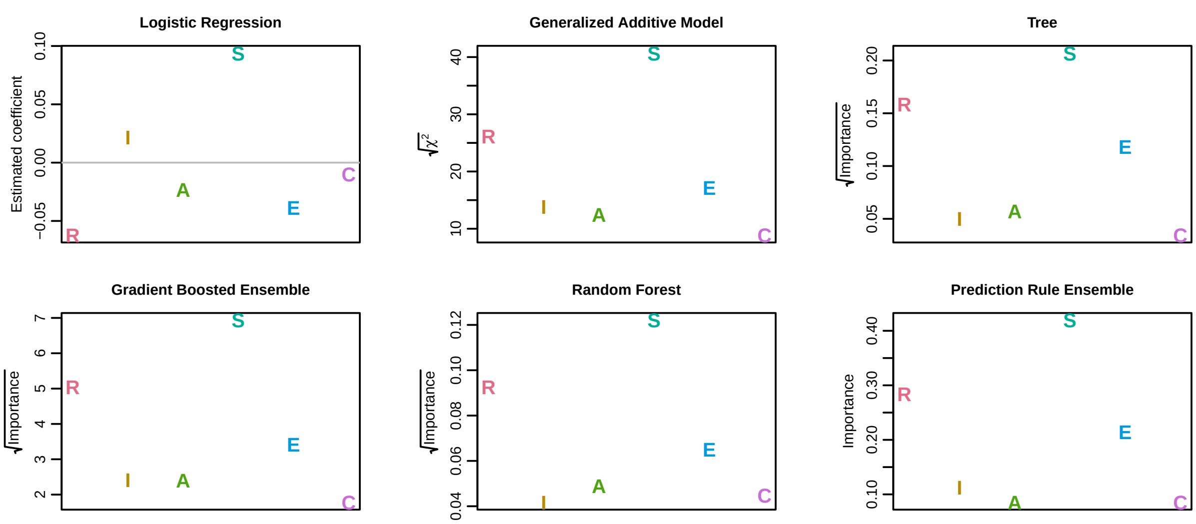
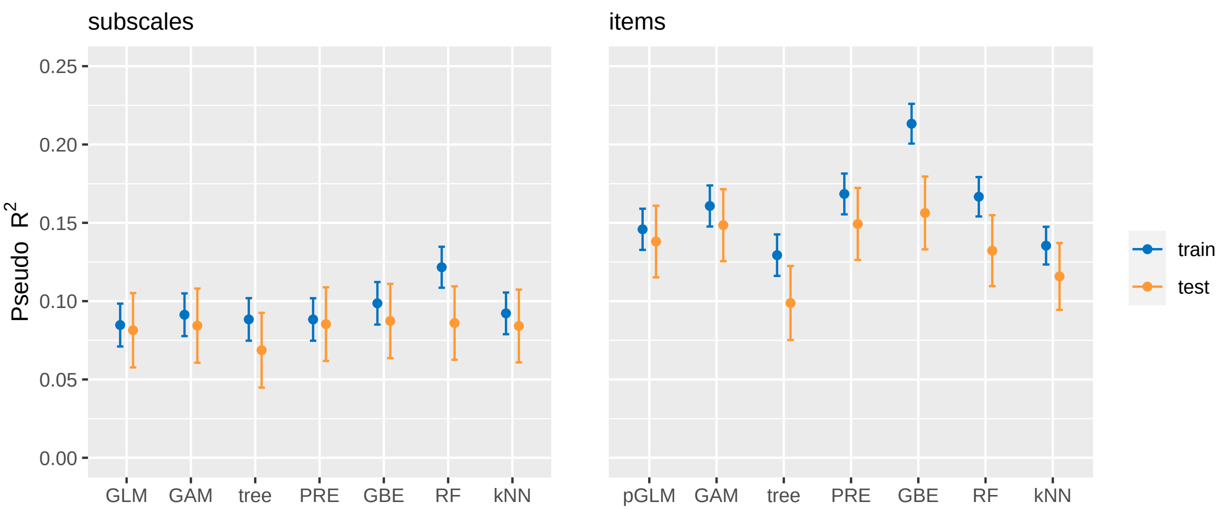


**Figure 5**

```
## Gather results for prediction with subscales
preds_train <- data.frame(bm_preds_train = rep(mean(train_y), times = length(train_ids)),
                          glm_preds_train_s, gam_preds_train_s,
                          ct_preds_train_s, pr_preds_train_s, gb_preds_train_s,
                          rf_preds_train_s, knn_preds_train_s)
preds_test <- data.frame(bm_preds_test = rep(mean(train_y), times = length(test_ids)),
                         glm_preds_test_s, gam_preds_test_s,
                         ct_preds_test_s, pr_preds_test_s, gb_preds_test_s,
                         rf_preds_test_s, knn_preds_test_s)
results <- data.frame(
  Brier_train = sapply(preds_train, function(x) mean((train_y - x)^2)),
  Brier_test = sapply(preds_test, function(x) mean((test_y - x)^2)))
results$R2_train <- 1 - results$Brier_train / results$Brier_train[1]
results$R2_test <- 1 - results$Brier_test / results$Brier_test[1]

tmp <- data.frame(
  R2 = c(results$R2_train[-1], results$R2_test[-1]),
  SE = c(sapply((preds_train[,-1] - train_y)^2,
              function(x) sd(x)/(sqrt(nrow(preds_train))*results$Brier_train[1])),
         sapply((preds_test[,-1] - test_y)^2,
              function(x) sd(x)/(sqrt(nrow(preds_test))*results$Brier_test[1]))),
  data = rep(c("train", "test"), each = nrow(results)-1),
  method = rep(c("GLM", "GAM", "tree", "PRE", "GBE", "RF", "kNN"), times = 2))
tmp$data <- factor(tmp$data, levels = c("train", "test"))
tmp$method <- factor(tmp$method, levels = c("GLM", "GAM", "tree", "PRE", "GBE", "RF", "kNN"))
tmp_s <- tmp

## Plot results
library("ggplot2")
p_subscales <- ggplot(tmp, aes(x = method, y = R2)) +
  geom_point(aes(color = data, fill = data),
             stat = "identity", position = position_dodge(0.4)) +
  ggtitle("subscales") + xlab("") + ylab(expression("Pseudo "~R^2)) +
  theme(legend.position = "none", plot.title = element_text(size = 11)) +
  scale_color_manual(values = c("#0073C2FF", "#ff9933")) + ylim(0, 0.25) +
  scale_fill_manual(values = c("#0073C2FF", "#ff9933")) +
  geom_errorbar(aes(color=data, ymin = R2-1.96*SE, ymax = R2+1.96*SE),
                width = .2, position = position_dodge(0.4))

## Gather results for prediction with items
preds_train <- data.frame(bm_preds_train = rep(mean(train_y), times = length(train_ids)),
```

```
                          glm_preds_train_i, gam_preds_train_i,
                          ct_preds_train_i, pr_preds_train_i, gb_preds_train_i,
                          rf_preds_train_i, knn_preds_train_i)
preds_test <- data.frame(bm_preds_test = rep(mean(train_y), times = length(test_ids)),
                          glm_preds_test_i, gam_preds_test_i,
                          ct_preds_test_i, pr_preds_test_i, gb_preds_test_i,
                         rf_preds_test_i, knn_preds_test_i)
results <- data.frame(
  Brier_train = sapply(preds_train, function(x) mean((train_y - x)^2)),
  Brier_test = sapply(preds_test, function(x) mean((test_y - x)^2)))
results$R2_train <- 1 - results$Brier_train / results$Brier_train[1]
results$R2_test <- 1 - results$Brier_test / results$Brier_test[1]

tmp <- data.frame(
  R2 = c(results$R2_train[-1], results$R2_test[-1]),
  SE = c(sapply((preds_train[,-1] - train_y)^2,
                function(x) sd(x)/(sqrt(nrow(preds_train))*results$Brier_train[1])),
          sapply((preds_test[,-1] - test_y)^2,
                function(x) sd(x)/(sqrt(nrow(preds_test))*results$Brier_test[1]))),
  data = rep(c("train", "test"), each = nrow(results)-1),
  method = rep(c("pGLM", "GAM", "tree", "PRE", "GBE", "RF", "kNN"), times = 2))
tmp$data <- factor(tmp$data, levels = c("train", "test"))
tmp$method <- factor(tmp$method, levels = c("pGLM", "GAM", "tree", "PRE", "GBE", "RF", "kNN"))

## Plot results
p_items <- ggplot(tmp, aes(x = method, y = R2)) +
  geom_point(aes(color = data, fill = data),
    stat = "identity", position = position_dodge(0.4)) + xlab("") + ylab(" ") +
  theme(axis.text.y=element_blank(), axis.ticks.y = element_blank(),
        plot.title = element_text(size = 11), legend.title = element_blank()) +
  ggtitle("items") + scale_color_manual(values = c("#0073C2FF", "#ff9933")) +
  scale_fill_manual(values = c("#0073C2FF", "#ff9933")) + ylim(0, .25) +
  geom_errorbar(aes(color=data, ymin = R2-1.96*SE, ymax = R2+1.96*SE),
                width = .2, position = position_dodge(0.4))
tmp_i <- tmp

library("gridExtra")
grid.arrange(p_subscales, p_items, ncol = 2, widths = c(6.75, 8))
```




```r
tmp_s <- cbind(tmp_s[tmp_s$data == "test", ], tmp_s[tmp_s$data == "train", ])
tmp_i <- cbind(tmp_i[tmp_i$data == "test", ], tmp_i[tmp_i$data == "train", ])
colnames(tmp_s) <- colnames(tmp_i) <- c("R2 (test)", "SE (test)", "data", "method",
                                        "R2 (train)", "SE (train)", "data", "method")
tmp_s$rank <- rank(1 - tmp_s[ , "R2 (test)"])
tmp_i$rank <- rank(1 - tmp_i[ , "R2 (test)"])
knitr::kable(tmp_s[ , c(4, 1, 2, 5, 6, 9)], row.names = FALSE, digits = 4L) ## scale scores
```

| method | R2 (test) | SE (test) | R2 (train) | SE (train) | rank |
|---|---|---|---|---|---|
| GLM | 0.0814 | 0.0121 | 0.0847 | 0.0070 | 6 |
| GAM | 0.0843 | 0.0121 | 0.0913 | 0.0070 | 4 |
| tree | 0.0686 | 0.0122 | 0.0883 | 0.0069 | 7 |
| PRE | 0.0853 | 0.0120 | 0.0883 | 0.0069 | 3 |
| GBE | 0.0873 | 0.0121 | 0.0986 | 0.0069 | 1 |
| RF | 0.0860 | 0.0120 | 0.1216 | 0.0067 | 2 |
| kNN | 0.0841 | 0.0119 | 0.0922 | 0.0068 | 5 |

```r
knitr::kable(tmp_i[ , c(4, 1, 2, 5, 6, 9)], row.names = FALSE, digits = 4L) ## item scores
```

| method | R2 (test) | SE (test) | R2 (train) | SE (train) | rank |
|---|---|---|---|---|---|
| pGLM | 0.1381 | 0.0117 | 0.1458 | 0.0067 | 4 |
| GAM | 0.1485 | 0.0117 | 0.1608 | 0.0067 | 3 |
| tree | 0.0988 | 0.0121 | 0.1294 | 0.0068 | 7 |
| PRE | 0.1493 | 0.0117 | 0.1684 | 0.0066 | 2 |
| GBE | 0.1563 | 0.0119 | 0.2133 | 0.0065 | 1 |
| RF | 0.1322 | 0.0116 | 0.1666 | 0.0064 | 5 |

| method | R2 (test) | SE (test) | R2 (train) | SE (train) | rank |
|---|---|---|---|---|---|
| kNN | 0.1158 | 0.0109 | 0.1354 | 0.0061 | 6 |

# Appendix B (ESM 2)

```r
data$age[data$age > 103] <- NA ## remove impossible ages
mean(data$age, na.rm = TRUE)
```

```
## [1] 33.0596
```

```r
sd(data$age, na.rm = TRUE)
```

```
## [1] 11.68003
```

```r
prop.table(table(data$gender)) ## 0=missing, 1=Male, 2=Female, 3=Other
```

```
##
##           0           1           2           3
## 0.001151224 0.320382062 0.673160290 0.005306423
```

```r
prop.table(table(data$married)) ## 0=missing, 1,2,3=Never, currently, previously married
```

```
##
##           0           1           2           3
## 0.005648193 0.582717249 0.328242764 0.083391794
```

```r
prop.table(table(data$urban)) ## 0=missing, 1=Rural, 2=Suburban, 3=Urban
```

```
##
##           0           1           2           3
## 0.007087223 0.217419459 0.372474952 0.403018366
```

```r
prop.table(table(data$race)) ## 0=missing, 1=Asian, 2=Arab, 3=Black,
```

```
##
##          0          1          2          3          4          5
## 0.01404853 0.19833432 0.01329304 0.07436188 0.60887162 0.09109060
```

```r
## 4=Indigenous Australian/Native American/White, 5=Other
## (A coding mistake resulted in aggregating options in category 4)
prop.table(table(data$engnat)) ## 0=missing, 1=Yes, 2=No
```

```
##
##           0           1           2
## 0.002248485 0.659813286 0.337938230
```

**Figure B1**

```r
par(mfrow = c(2, 3))
for (i in c("Real", "Inve", "Arti", "Soci" , "Ente", "Conv")) {
  dens <- density(data[ , i])
  plot(dens, main = "", xlab = i)
  polygon(dens, col="lightblue", border="black")
}
```

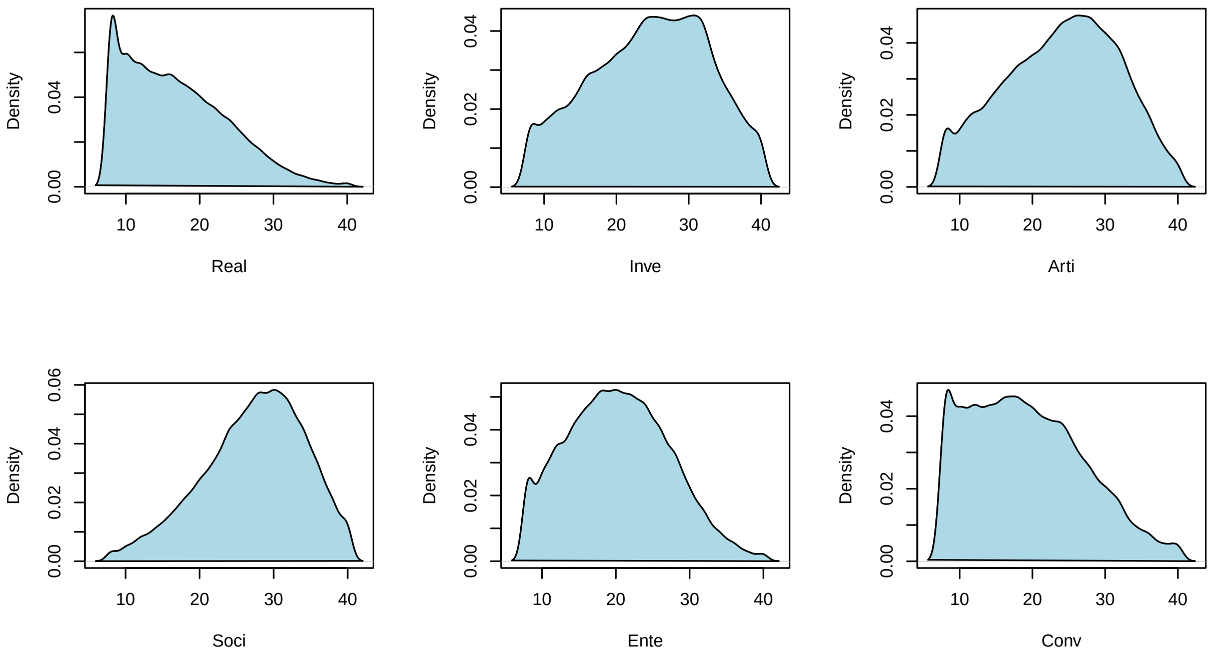



**Table B1**

```r
tab <- round(cor(data[ , c("Real", "Inve", "Arti", "Soci" , "Ente", "Conv")]),
             digits = 3L)
knitr::kable(tab)
```

| | Real | Inve | Arti | Soci | Ente | Conv |
|---|---|---|---|---|---|---|
| Real | 1.000 | 0.332 | 0.182 | 0.049 | 0.305 | 0.460 |
| Inve | 0.332 | 1.000 | 0.329 | 0.141 | 0.016 | 0.065 |
| Arti | 0.182 | 0.329 | 1.000 | 0.290 | 0.253 | -0.056 |
| Soci | 0.049 | 0.141 | 0.290 | 1.000 | 0.356 | 0.124 |
| Ente | 0.305 | 0.016 | 0.253 | 0.356 | 1.000 | 0.464 |

| | Real | Inve | Arti | Soci | Ente | Conv |
|---|---|---|---|---|---|---|
| Conv | 0.460 | 0.065 | -0.056 | 0.124 | 0.464 | 1.000 |

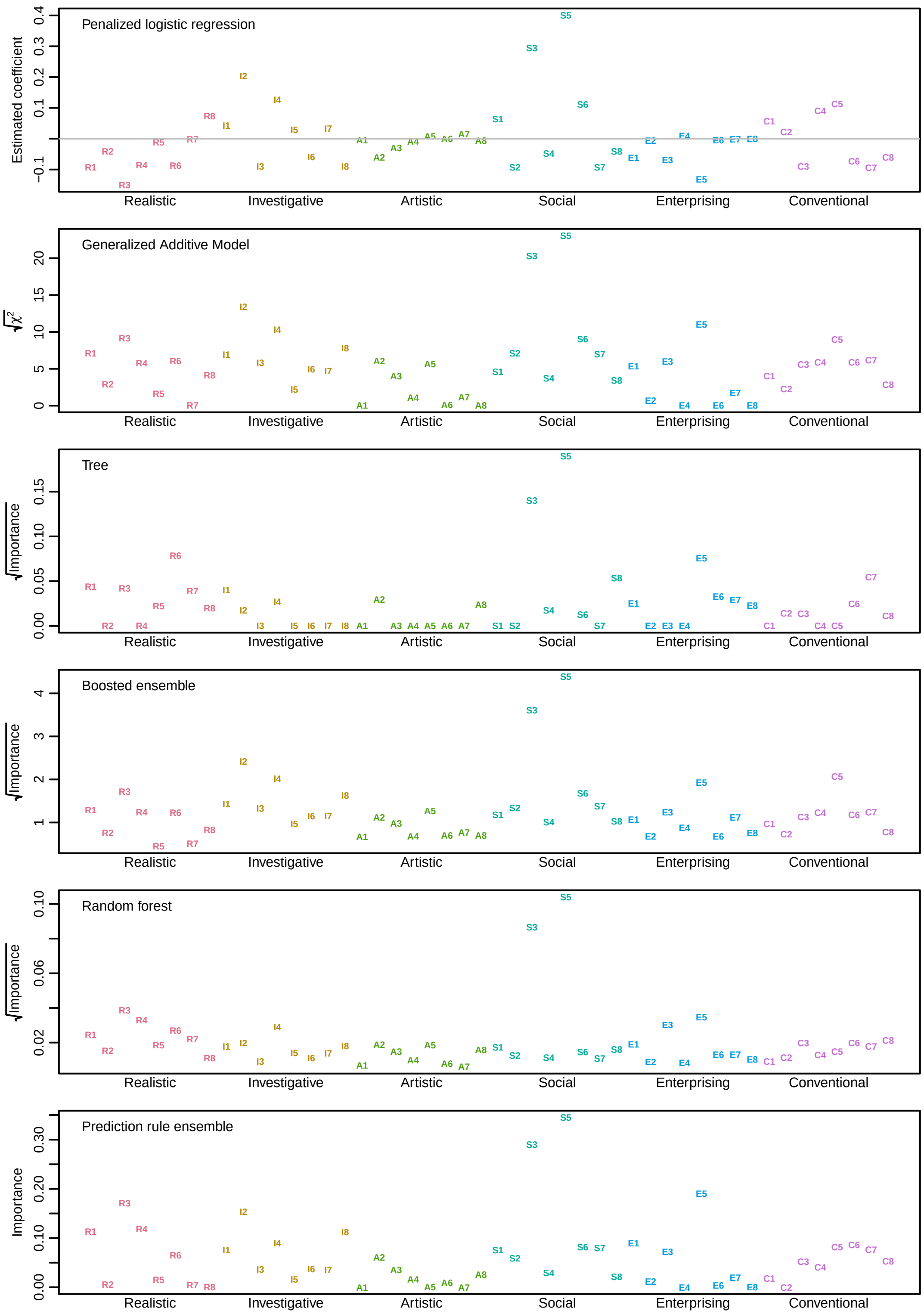
## Appendix C (ESM 3)

**Figure C1**

```
par(mar = c(1.5, 4, 0.2, 2), mgp = c(1.5, .5, 0), tck = -0.05)
par(mfrow = c(6, 1))

library("glmnet")
plot(coef(glmod_i)[-1], xaxt = "n", ylab = "Estimated coefficient",
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = " ", cex.main = .7)
text(coef(glmod_i)[-1], labels = varnames_i, cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
abline(0, 0, col = "grey")
legend("topleft", legend = "Penalized logistic regression", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
                              "Enterprising", "Conventional"), cex.axis = .7)

library("mgcv")
load(file = "GAM_items.Rda")
sum <- summary(gamod_i)
plot(sqrt(sum$chi.sq), xaxt = "n", ylab = expression(sqrt(chi^2)),
     main = " ", cex.main = .7,
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ")
text(sqrt(sum$chi.sq), labels = varnames_i, cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
legend("topleft", legend = "Generalized Additive Model", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
                              "Enterprising", "Conventional"), cex.axis = .7)

ct6 <- cforest(ct_form,
               data = data[train_ids , ], ntree = 1L, mtry = 6,
               perturb = list(replace = FALSE, fraction = 1L),
               control = ctree_control(maxdepth = 6))
imps <- varimp(gettree(ct6), risk = "loglik")
imp_names <- names(imps)
imps <- c(imps, rep(0, times = 48 - length(imps)))
names(imps) <- c(imp_names, varnames_i[!varnames_i %in% imp_names])
plot(sqrt(imps[varnames_i]), xaxt = "n", ylab = expression(sqrt(Importance)),
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     main = " ", cex.main = .7)
```

```
text(sqrt(imps[varnames_i]), labels = varnames_i, cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
legend("topleft", legend = "Tree", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
                              "Enterprising", "Conventional"), cex.axis = .7)

load(file = "gb_i_summary.Rda")
imps <- sum_i[match(varnames_i, sum_i$var), ]
plot(sqrt(imps$rel.inf), xaxt = "n", main = " ",
     ylab = expression(sqrt(Importance)), cex.main = .7,
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ")
text(sqrt(imps$rel.inf), labels = imps$var, cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
legend("topleft", legend = "Boosted ensemble", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
                              "Enterprising", "Conventional"), cex.axis = .7)

imps <- ranger::importance(rf_i)
plot(sqrt(imps), xaxt = "n", main = " ",
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ",
     ylab = expression(sqrt(Importance)), cex.main = .7)
text(sqrt(imps), labels = names(imps), cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
legend("topleft", legend = "Random forest", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
                              "Enterprising", "Conventional"), cex.axis = .7)

imps <- pre::importance(pr_i, cex.axis = .7, plot = FALSE)$varimps
zero_vars <- varnames_i[!varnames_i %in% imps[ , 1]]
imps <- rbind(imps, data.frame(varname = zero_vars,
                                imp = rep(0, times = length(zero_vars))))
imps <- imps[match(varnames_i, imps$varname), ]
plot(imps$imp, xaxt = "n", main = " ",
     ylab = "Importance",
     col = "white", cex.lab = .7, cex.axis = .7, xlab = " ", cex.main = .7)
text(imps$imp, labels = imps$varname, cex = .5,
     col = rep(qualitative_hcl(6), each = 8), font = 2)
legend("topleft", legend = "Prediction rule ensemble", cex = .7, bty = "n")
axis(1, 4.5 + c(0:5)*8, tick = FALSE, padj = -1.5,
     labels = c("Realistic", "Investigative", "Artistic", "Social" ,
```

```
                    "Enterprising", "Conventional"), cex.axis = .7)
```



## References


Fokkema, M. (2020). Fitting prediction rule ensembles with R package pre. *Journal of Statistical Software*, *92*(1), 1–30.

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, *33*(1), 1.

Greenwell, B., Boehmke, B., Cunningham, J., & Developers, G. (2020). *gbm: Generalized boosted regression models*. https://CRAN.R-project.org/package=gbm

Hothorn, T., Hornik, K., & Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, *15*(3), 651–674. https://doi.org/10.1198/106186006X133933

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S* (4th Edition). Springer. https://www.stats.ox.ac.uk/pub/MASS4/

Wood, S. N. (2017). *Generalized additive models: An introduction with R*. CRC press.

Wright, M. N., & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, *77*(1), 1–17. https://doi.org/10.18637/jss.v077.i01