analysis for Bansal et al.

2023-12-02

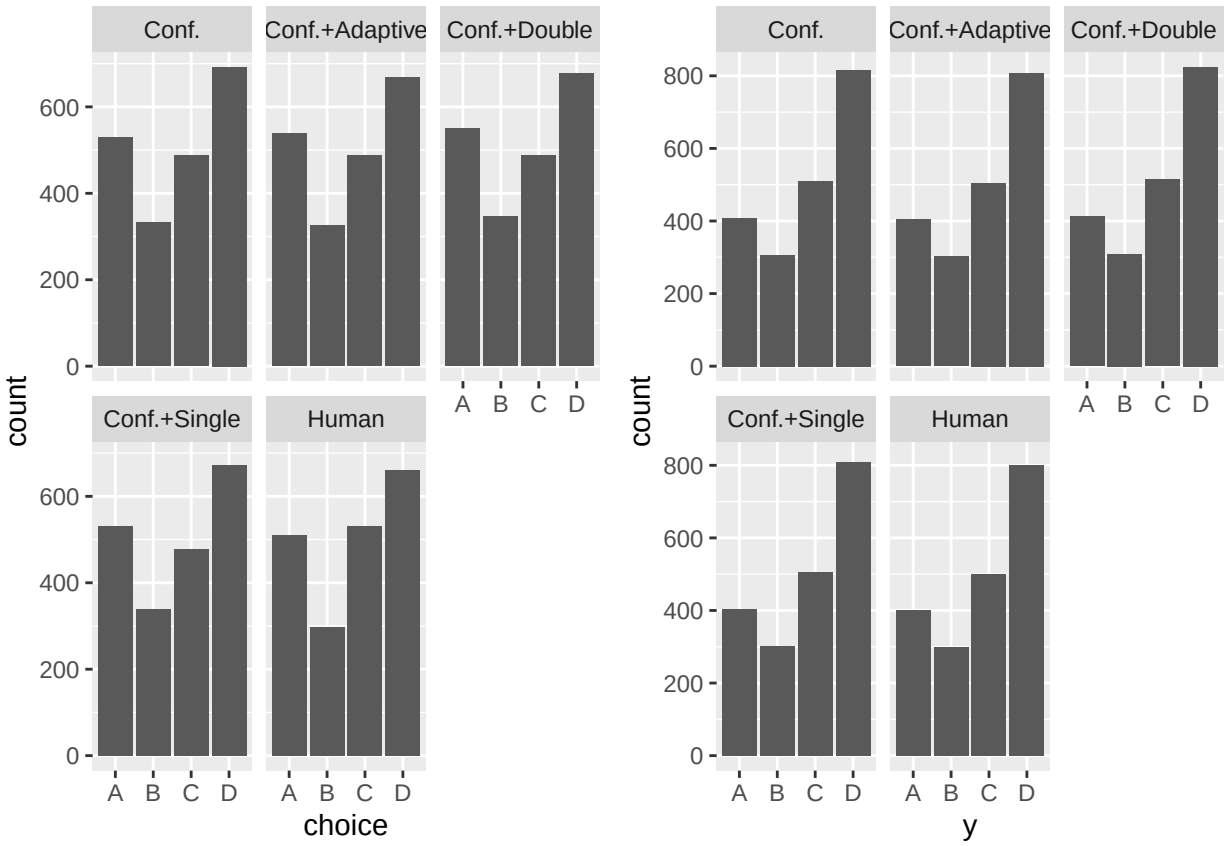
In this document, we run the reliance framework analysis for LSAT task and its corresponding explanations in Bansal et al. We estimate the behavioral agents' joint behavioral $\pi(\theta, v, a^b)$ by a statistical model fitted on the experiment data. We first show the results using approximation of rational benchmark and the mis-reliant rational benchmark with overfitting to the empirical distribution and then show the results using approximation with the discretized signals generated by the K-Means model.

```
task_name = "lsat"  # "lsat" | "beer" | "amzbook"
bonus =1 # 0.3  # 0.3 | 0.05
```

```
raw_data = read.csv("decision-result-filter.csv") %>% filter(task == task_name)
```

```
choice_plot = raw_data %>%
  ggplot() +
  geom_bar(aes(x = choice)) +
  facet_wrap(vars(condition))

y_plot = raw_data %>%
  ggplot() +
  geom_bar(aes(x = y)) +
  facet_wrap(vars(condition))
plot_grid(choice_plot, y_plot)
```

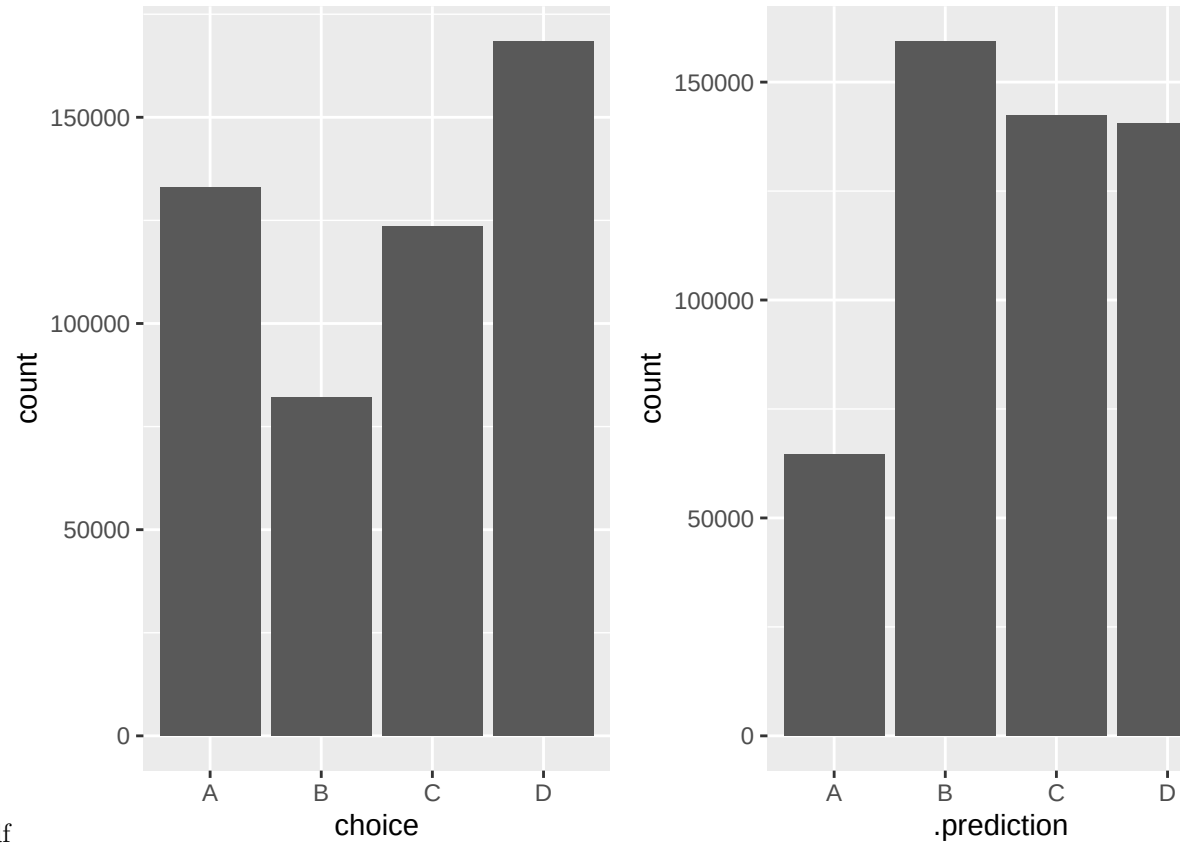



```
prior.choice <- brm(data = raw_data, family = "categorical",
                    bf(choice ~ 1),
                    sample_prior = "only",
                    iter = 3000, warmup = 500, chains = 4, cores = 4)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

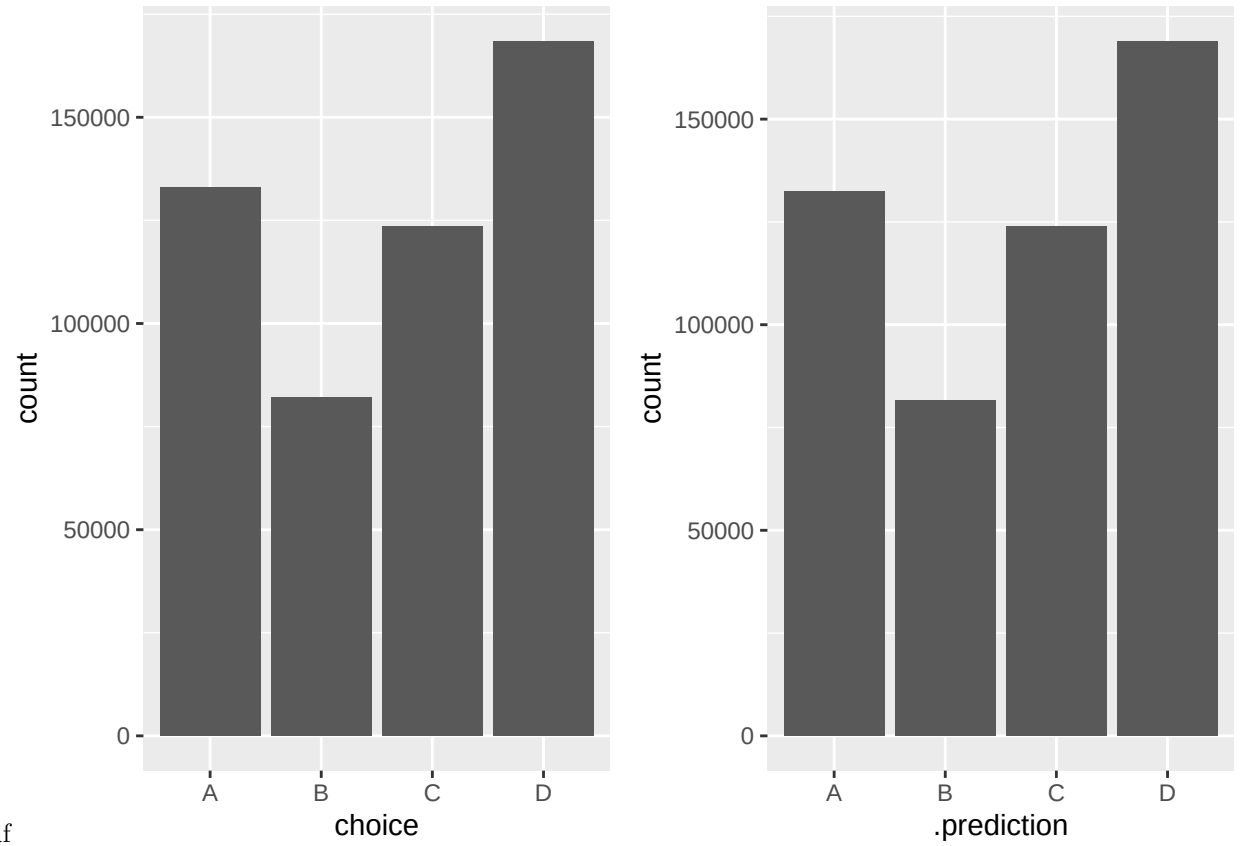
```
pred = raw_data %>%
  add_predicted_draws(prior.choice, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar()
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar()
plot_grid(obs_plot, model_plot)
```



prior choice model-1.pdf

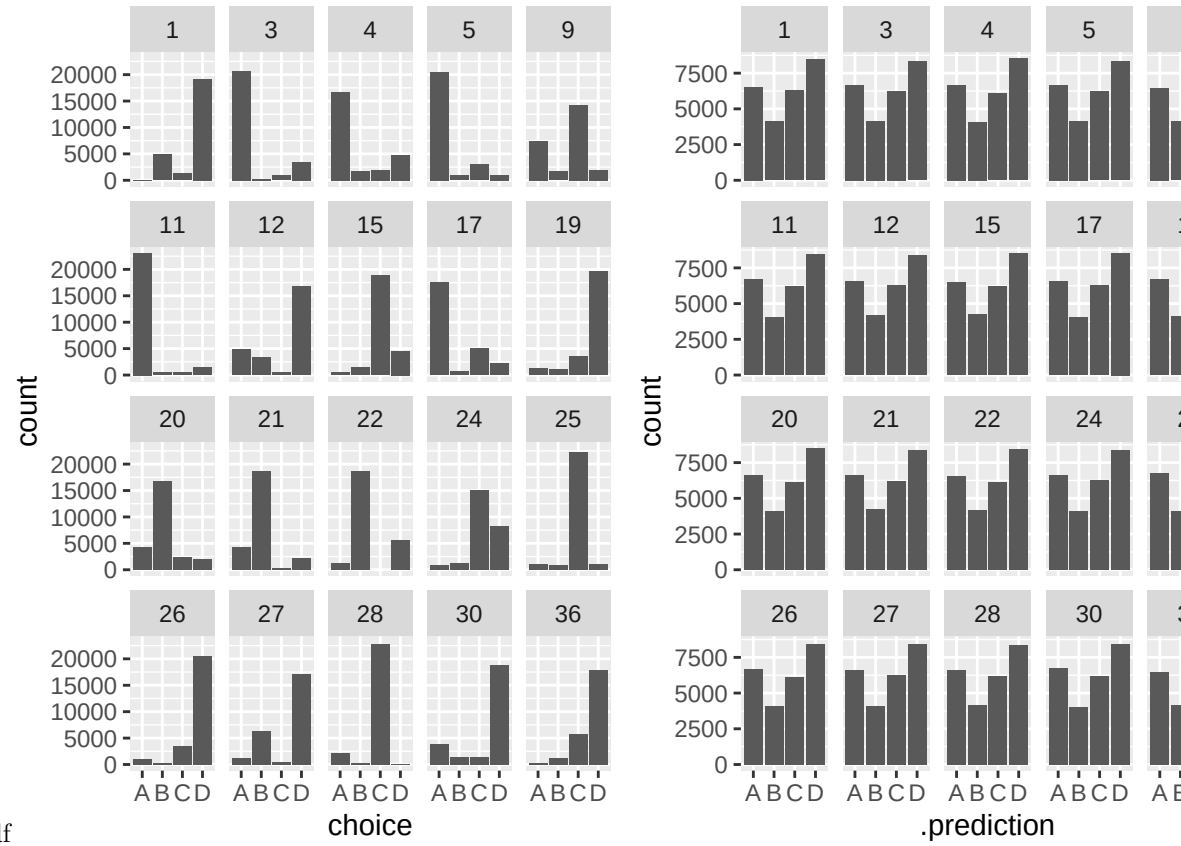
```
m.choice = brm(data = raw_data, family = "categorical",
                       bf(choice ~ 1),
                       iter = 3000, warmup = 500, chains = 4, cores = 4,
               file = "./models/m_choice_mdl")
```

```
pred = raw_data %>%
  add_predicted_draws(m.choice, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar()
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar()
plot_grid(obs_plot, model_plot)
```



choice model-1.pdf

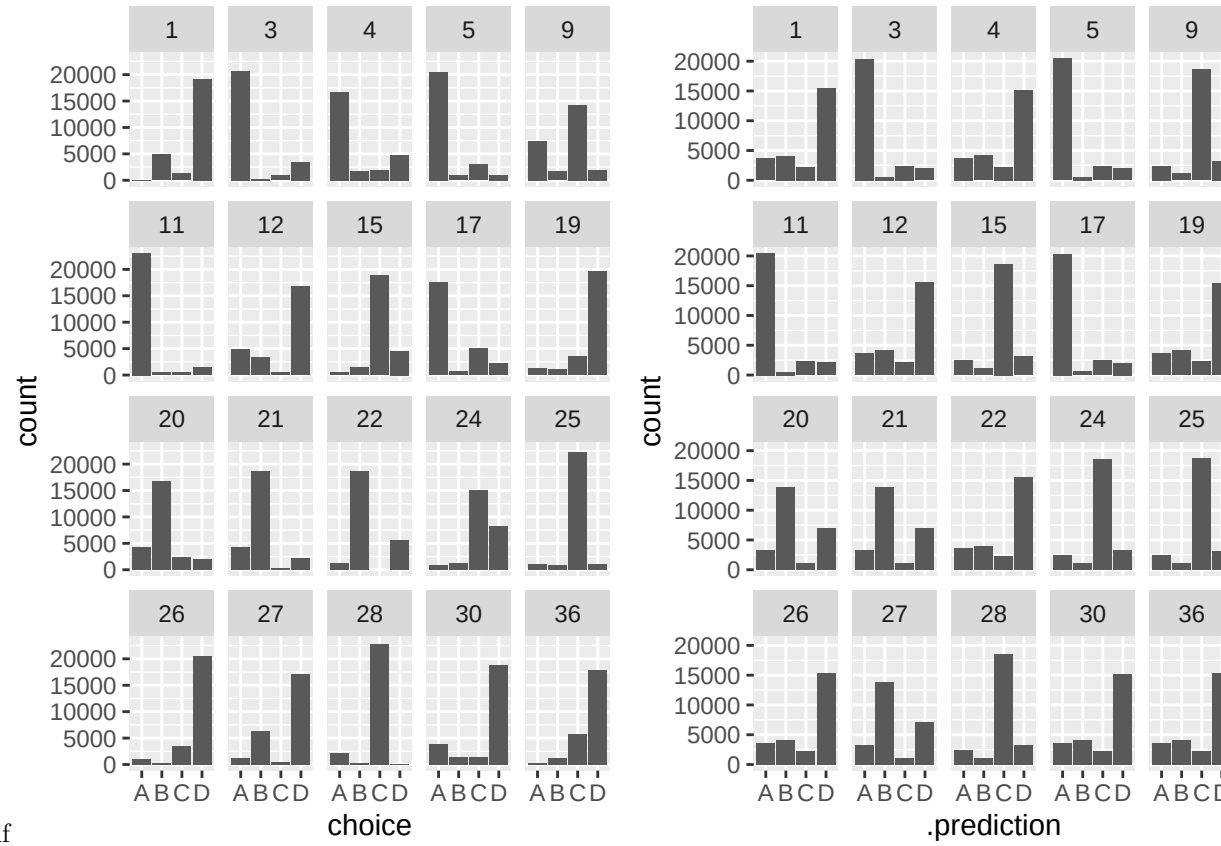
```
pred = raw_data %>%
  add_predicted_draws(m.choice, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



facet choice model-1.pdf

```
m.choice_y = brm(data = raw_data, family = "categorical",
                 bf(choice ~ y),
                 iter = 3000, warmup = 500, chains = 4, cores = 4,
                 file = "./models/m_choice_y_mdl")
```

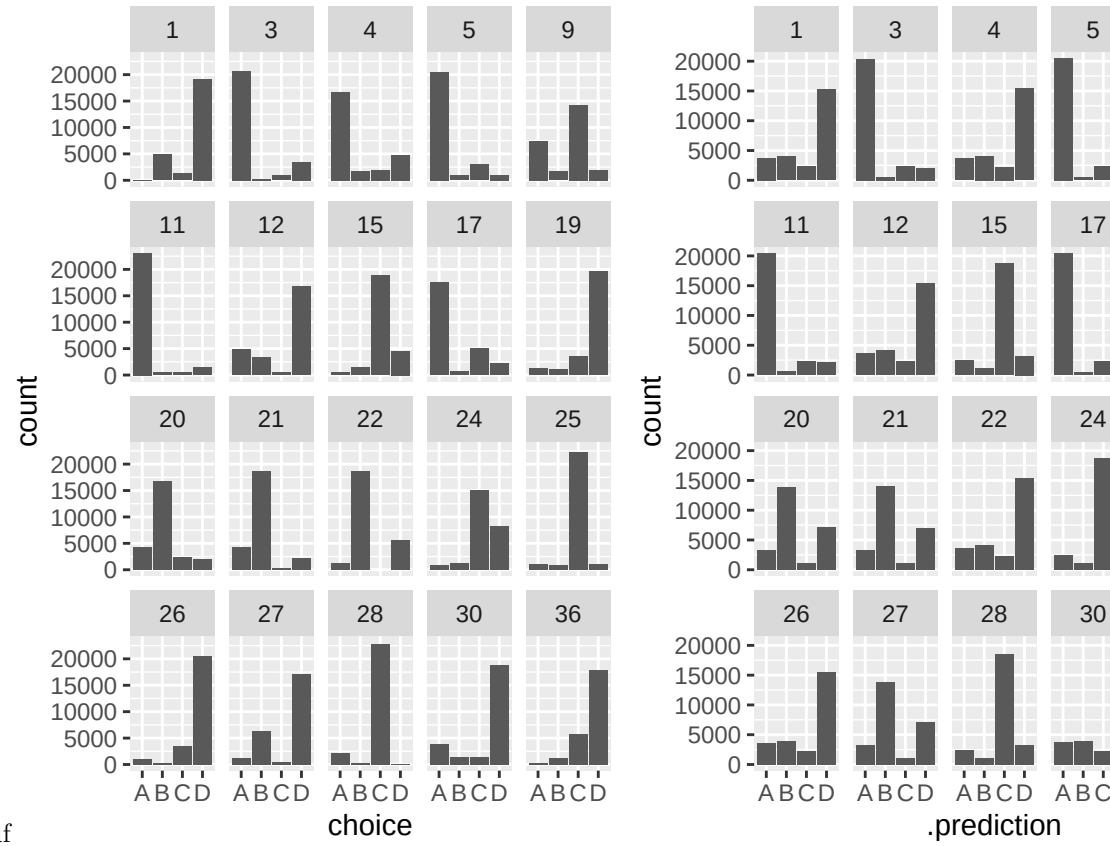
```
pred = raw_data %>%
  add_predicted_draws(m.choice_y, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



choice y model-1.pdf

```
m.choice_y_condition = brm(data = raw_data, family = "categorical",
                      bf(choice ~ y + condition),
                      iter = 3000, warmup = 500, chains = 4, cores = 4,
                      file = "./models/m_choice_y_condition_mdl")
```

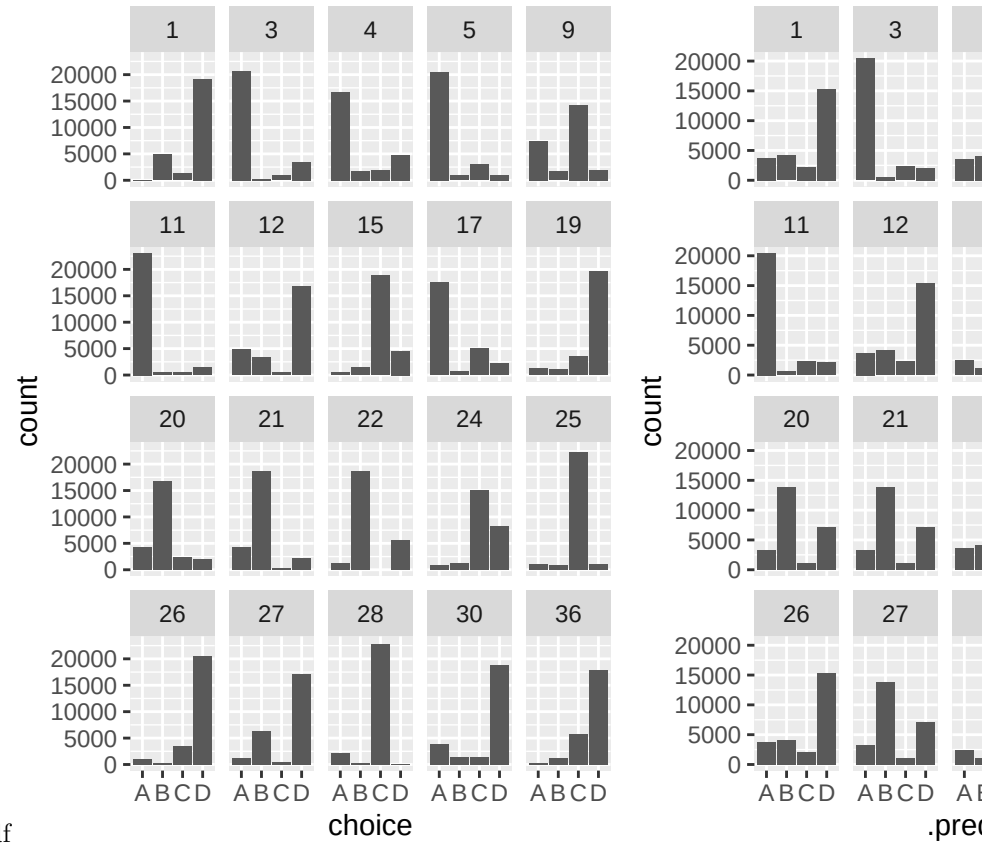
```
pred = raw_data %>%
  add_predicted_draws(m.choice_y_condition, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



choice y condition model-1.pdf

```
m.choice_y_condition_interaction = brm(data = raw_data, family = "categorical",
                       bf(choice ~ y * condition),
                       iter = 3000, warmup = 500, chains = 4, cores = 4,
                       file = "./models/m_choice_y_condition_interaction_mdl")
```

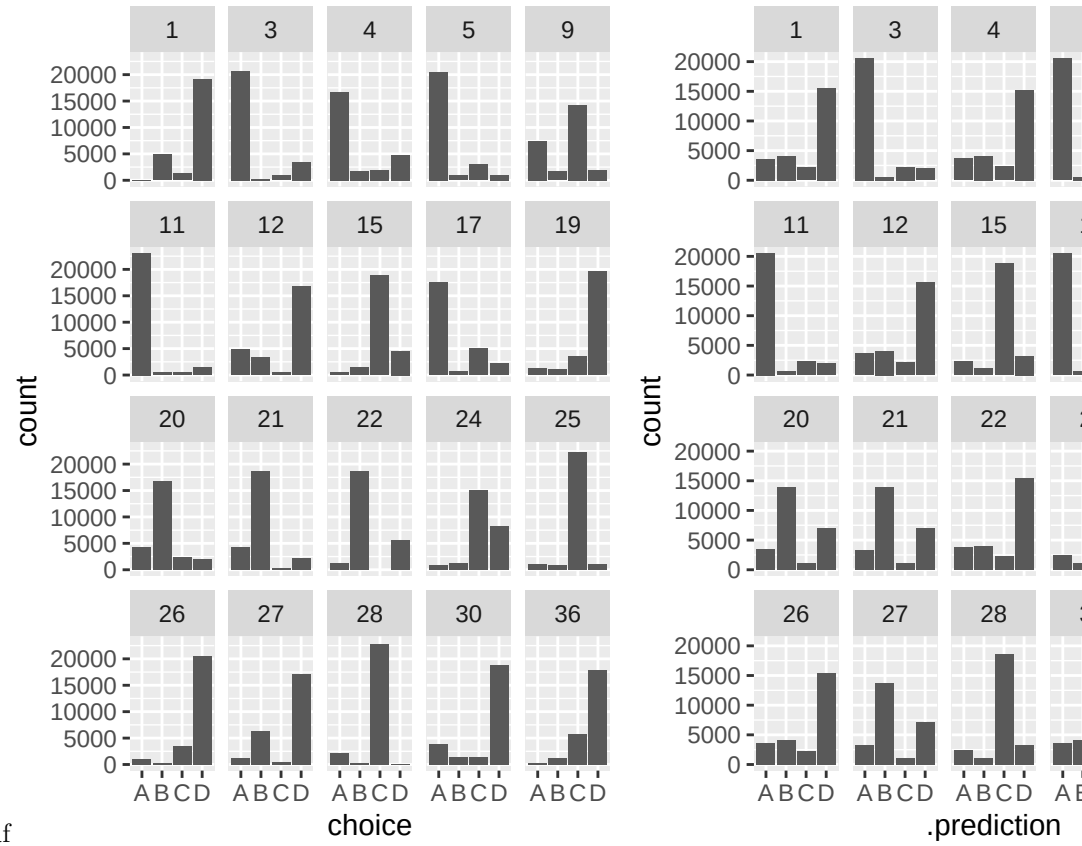
```
pred = raw_data %>%
  add_predicted_draws(m.choice_y_condition_interaction, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



choice y condition interaction model-1.pdf

```
m.choice_y_condition_re = brm(data = raw_data, family = "categorical",
                      bf(choice ~ y * condition + (1 | assignmentId)),
                      iter = 3000, warmup = 500, chains = 4, cores = 4,
                      file = "./models/m_choice_y_condition_re_mdl")
```

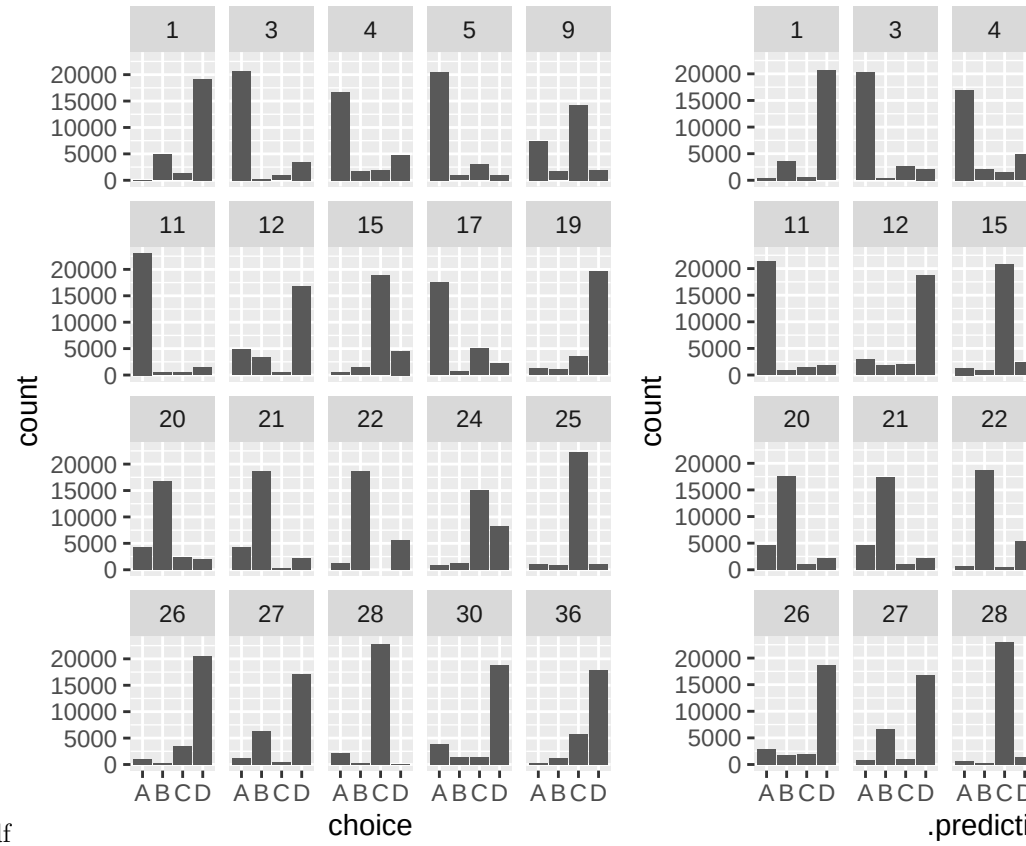
```
pred = raw_data %>%
  add_predicted_draws(m.choice_y_condition_re, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



choice y condition re model-1.pdf

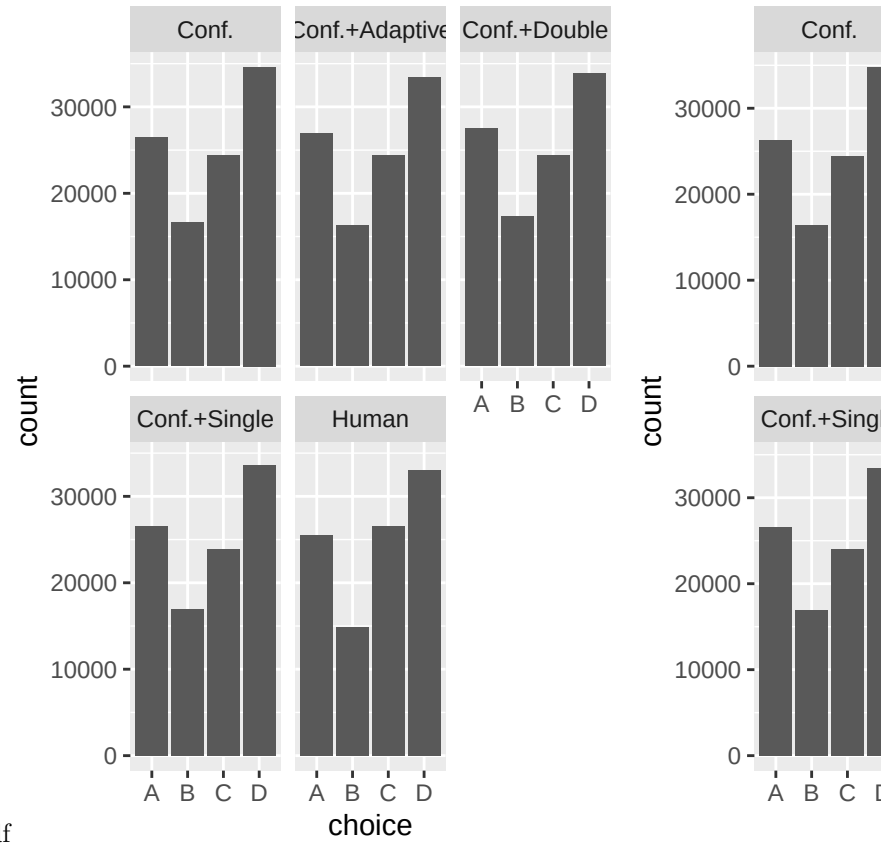
```
m.choice_y_condition_pred_re = brm(data = raw_data, family = "categorical",
                      bf(choice ~ y * condition + pred + pred2 + (1 | assignmentId)),
                      iter = 3000, warmup = 500, chains = 4, cores = 4,
                      file = "./models/m_choice_y_condition_pred_re_mdl")
```

```
pred = raw_data %>%
  add_predicted_draws(m.choice_y_condition_pred_re, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(questionId))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(questionId))
plot_grid(obs_plot, model_plot)
```



choice y condition pred re model-1.pdf

```
pred = raw_data %>%
  add_predicted_draws(m.choice_y_condition_pred_re, ndraws = 50)
obs_plot = pred %>% ggplot(aes(x = choice)) + geom_bar() + facet_wrap(vars(condition))
model_plot = pred %>% ggplot(aes(x = .prediction)) + geom_bar() + facet_wrap(vars(condition))
plot_grid(obs_plot, model_plot)
```



choice y condition pred re model on condition-1.pdf

```
meta_data = raw_data %>% group_by(questionId, condition, y, pred, conf, conf2, pred2) %>% summarise()
```

```
## `summarise()` has grouped output by 'questionId', 'condition', 'y', 'pred',
## 'conf', 'conf2'. You can override using the `.groups` argument.
```

```
pred_human_data = meta_data %>%
  filter(condition == "Human") %>%
  add_predicted_draws(m.choice_y_condition_pred_re, ndraws = 1000, re_formula = NA)
human_predictions = pred_human_data %>%
  rename(human_pred = .prediction) %>%
  ungroup() %>%
  select(-condition, -.row, -.chain, -.iteration)
exp_data = meta_data %>%
  filter(condition != "Human") %>%
  left_join(human_predictions,
            by = join_by(questionId, y, pred, conf, conf2, pred2)) %>%
  rename(drawId = .draw) %>%
  add_predicted_draws(m.choice_y_condition_pred_re, ndraws = 1, re_formula = NA) %>%
  rename(choice = .prediction, drawId2 = .draw) %>%
  ungroup() %>%
  select(-.row, -.chain, -.iteration)
```

```
## Warning in left_join(., human_predictions, by = join_by(questionId, y, pred, : Detected an unexpecte
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 1 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
con <- file('./task-lsat.json', "r")
task_data <- ldply(fromJSON(con), data.frame)
task_data = task_data[!duplicated(task_data$id), ]
task_data = task_data %>%
  unite("text_signal", choices.A:question, remove = FALSE)
```

```r
payoff = function(action, state) {
  (action == state) * bonus
}
expected_payoff = function(action, states) {
  return(mean((action == states) * bonus))
}
```

```r
corpus = tm::Corpus(tm::VectorSource(task_data$text_signal))
corpus.cleaned <- tm::tm_map(corpus, tm::removeWords, tm::stopwords('english')) # Removing stop-words
```

```
## Warning in tm_map.SimpleCorpus(corpus, tm::removeWords,
## tm::stopwords("english")): transformation drops documents
```

```r
corpus.cleaned <- tm::tm_map(corpus, tm::stemDocument, language = "english") # Stemming the words
```

```
## Warning in tm_map.SimpleCorpus(corpus, tm::stemDocument, language = "english"):
## transformation drops documents
```

```r
corpus.cleaned <- tm::tm_map(corpus.cleaned, tm::stripWhitespace) # Trimming excessive whitespaces
```

```
## Warning in tm_map.SimpleCorpus(corpus.cleaned, tm::stripWhitespace):
## transformation drops documents
```

```r
tdm <- tm::DocumentTermMatrix(corpus.cleaned)
tdm.tfidf <- tm::weightTfIdf(tdm)
tdm.tfidf <- tm::removeSparseTerms(tdm.tfidf, 0.999)
tfidf.matrix <- as.matrix(tdm.tfidf)
```

```r
predict.kmeans <- function(object, newdata){
  centers <- object$centers
  n_centers <- nrow(centers)
  dist_mat <- as.matrix(dist(rbind(centers, newdata)))
  dist_mat <- dist_mat[-seq(n_centers), seq(n_centers)]
  list(cluster = max.col(-dist_mat), total_error = sum(apply(dist_mat, 1, function(x) min(x))))
}

number_of_partition = 10
partition_size = nrow(tfidf.matrix) / number_of_partition
best.K = -1
best.test_sd = Inf
best.benchmark = c()
for (K in (2:17)) {
  benchmark = c()
  for (i in seq(1, nrow(tfidf.matrix), partition_size)) {
    test_set = tfidf.matrix[i:(i + partition_size - 1),]
    training_set = tfidf.matrix[-(i:(i + partition_size - 1)),]
    clustering.kmeans <- kmeans(training_set, K)
    cluster_number = predict(clustering.kmeans, test_set)$cluster
    test_questionId = task_data$id[i:(i + partition_size - 1)]
```

```r
    test_task_data = task_data %>% filter(id %in% test_questionId) %>% mutate(cluster = cluster_number)
    test_human_predictions = human_predictions %>%
      filter(questionId %in% test_questionId) %>%
      left_join(test_task_data %>% select(questionId = id, cluster), by = c("questionId"))

    train_task_data = task_data %>% filter(!(id %in% test_questionId)) %>% mutate(cluster = clustering.
    train_human_predictions = human_predictions %>%
      filter(!(questionId %in% test_questionId)) %>%
      left_join(train_task_data %>% select(questionId = id, cluster), by = c("questionId"))

    test_rational_action = train_human_predictions %>%
      rbind(test_human_predictions) %>%
      mutate(human_pred = as.character(human_pred)) %>%
      group_by(human_pred, pred, pred2, cluster) %>%
      mutate(human_payoff = payoff(human_pred, y),
             pred_payoff = payoff(pred, y),
             pred2_payoff = payoff(pred2, y)) %>%
      summarise(human_payoff = mean(human_payoff),
             pred_payoff = mean(pred_payoff),
             pred2_payoff = mean(pred2_payoff))
    benchmark = c(benchmark, (test_human_predictions %>%
      left_join(test_rational_action, by = c("pred", "pred2", "human_pred", "cluster")) %>%
      mutate(pred_better = ifelse(pred_payoff >= pred2_payoff, pred, pred2),
           pred_better_payoff = ifelse(pred_payoff >= pred2_payoff, pred_payoff, pred2_payoff)) %>%
      mutate(rational_benchmark_action = ifelse(pred_better_payoff > human_payoff, pred_better, human_p
      mutate(rational_benchmark = (rational_benchmark_action == y) * bonus) %>%
      summarise(rational_benchmark = mean(rational_benchmark)))$rational_benchmark)
  }
  if (best.test_sd > sd(benchmark)) {
    best.test_sd = sd(benchmark)
    best.benchmark = benchmark
    best.K = K
  }
}
```

```
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
```

```
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
```

```
best.K
```

```
## [1] 12
```

```
clustering.kmeans <- kmeans(tfidf.matrix, best.K)
```

```
task_data = task_data %>% mutate(cluster = clustering.kmeans$cluster)
exp_data = exp_data %>% left_join(task_data %>% select(questionId = id, cluster))
```

```
## Joining with `by = join_by(questionId)`
```

```
human_predictions = human_predictions %>% left_join(task_data %>% select(questionId = id, cluster))
```

```
## Joining with `by = join_by(questionId)`
```

```
rational_prior_action = ifelse(expected_payoff(human_predictions$human_pred, human_predictions$y) >
                                 max(expected_payoff(human_predictions$pred, human_predictions$y),
                               "human", "pred")
rational_prior_action
```

```
## [1] "pred"
```

Approximating by overfitting to the empirical distribution

```
rational_action = human_predictions %>%
  mutate(human_pred = as.character(human_pred)) %>%
  group_by(human_pred, pred, pred2, questionId) %>%
  mutate(human_payoff = payoff(human_pred, y),
            pred_payoff = payoff(pred, y),
            pred2_payoff = payoff(pred2, y)) %>%
  summarise(human_payoff = mean(human_payoff),
            pred_payoff = mean(pred_payoff),
            pred2_payoff = mean(pred2_payoff))
```

```
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
```

```
exp_data_with_reliance = exp_data %>%
  mutate(human_pred = as.character(human_pred)) %>%
  mutate(choice = as.character(choice)) %>%
  mutate(behavioral_payoff = (choice == y) * bonus) %>%
  left_join(rational_action) %>%
  # mutate(pred_better = pred,
  #        pred_better_payoff = pred_payoff) %>%
  mutate(pred_better = ifelse(pred_payoff >= pred2_payoff, pred, pred2),
         pred_better_payoff = ifelse(pred_payoff >= pred2_payoff, pred_payoff, pred2_payoff)) %>%
  group_by(drawId, drawId2, condition) %>%
  arrange(desc(pred_better_payoff - human_payoff), .by_group = TRUE) %>%
  mutate(sort_id = row_number()) %>%
  mutate(is_relying = ((choice == pred) | (choice == pred2)) & (choice != human_pred)) %>%
  mutate(reliance = sum(is_relying)) %>%
  ungroup() %>%
  mutate(action = ifelse(sort_id <= reliance,
                         pred_better,
                         human_pred)) %>%
  mutate(misreliant_payoff = (action == y) * bonus) %>%
  mutate(rational_benchmark_action = ifelse(pred_better_payoff > human_payoff, pred_better, human_pred)
  mutate(rational_benchmark = (rational_benchmark_action == y) * bonus) %>%
  mutate(rational_rl = ((rational_benchmark_action == pred) |
                                  (rational_benchmark_action == pred2)) &
           (rational_benchmark_action != human_pred)) %>%
  rowwise() %>%
  mutate(rational_baseline_action = pred) %>%
  mutate(rational_baseline = (rational_baseline_action == y) * bonus) %>%
  mutate(rational_baseline2_action = human_pred) %>%
  mutate(rational_baseline2 = (rational_baseline2_action == y) * bonus)
```

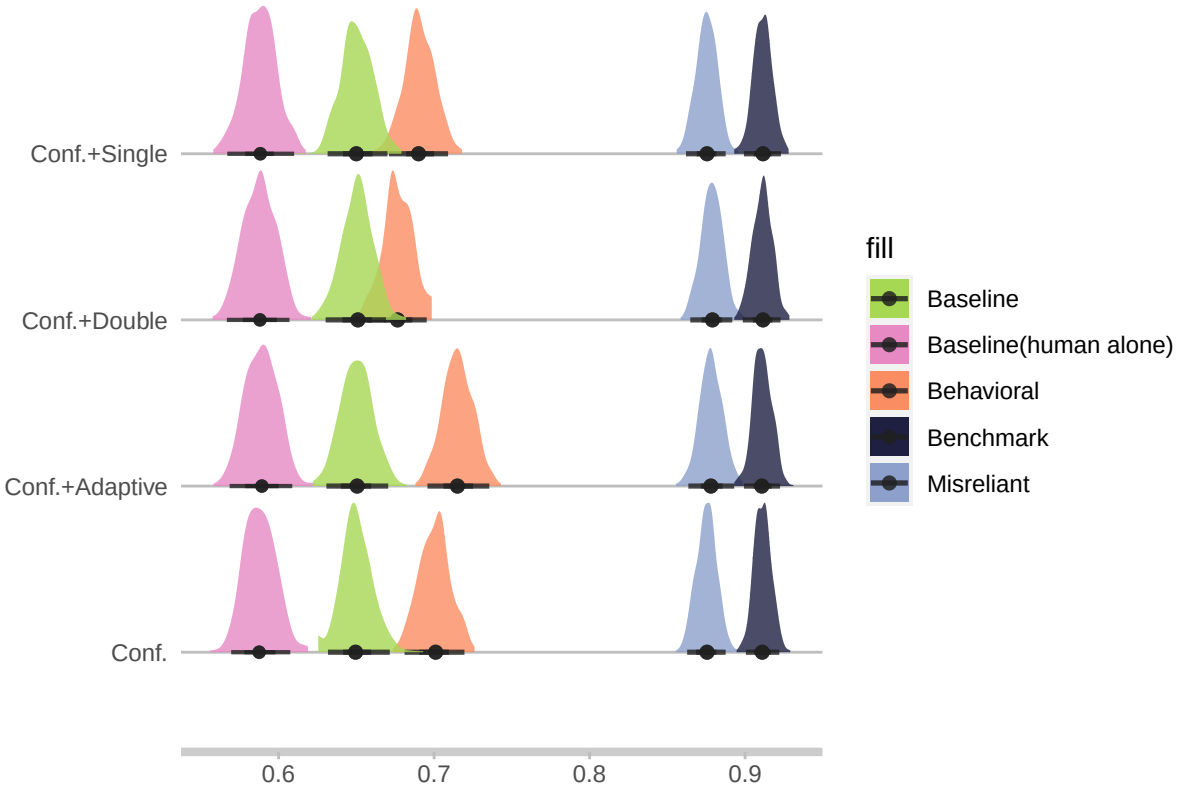
```
## Joining with `by = join_by(questionId, pred, pred2, human_pred)`
```

```
sample_size = 100 * 20
n_round = 500
results = data.frame()
for (i in 1:n_round) {
  results = exp_data_with_reliance %>%
    group_by(condition) %>%
    sample_n(sample_size) %>%
    # group_by(condition) %>%
```

```
    summarise(behavioral_payoff = mean(behavioral_payoff),
              misreliant_payoff = mean(misreliant_payoff),
              benchmark_payoff = mean(rational_benchmark),
              baseline_payoff = mean(rational_baseline),
              baseline2_payoff = mean(rational_baseline2),
              reliance = mean(reliance),
              rational_reliance = mean(rational_rl)) %>%
    rbind(results)
}
results
```

```
## # A tibble: 2,000 x 8
##    condition      behavioral_payoff misreliant_payoff benchmark_payoff
##    <chr>                      <dbl>             <dbl>            <dbl>
##  1 Conf.                      0.688             0.868            0.908
##  2 Conf.+Adaptive             0.710             0.874            0.907
##  3 Conf.+Double               0.680             0.889            0.918
##  4 Conf.+Single               0.678             0.871            0.91
##  5 Conf.                      0.689             0.872            0.910
##  6 Conf.+Adaptive             0.706             0.885            0.920
##  7 Conf.+Double               0.684             0.884            0.914
##  8 Conf.+Single               0.686             0.872            0.910
##  9 Conf.                      0.712             0.864            0.911
## 10 Conf.+Adaptive             0.724             0.894            0.918
## # i 1,990 more rows
## # i 4 more variables: baseline_payoff <dbl>, baseline2_payoff <dbl>,
## #   reliance <dbl>, rational_reliance <dbl>
```

```
colors <- c("Baseline" = "#a6d854", "Baseline(human alone)" = "#e78ac3", "Benchmark" = "#1F2041", "Beha
ggplot() +
  stat_slabinterval(data = results, aes(y = condition, x = behavioral_payoff, fill = "Behavioral"), alp
  stat_slabinterval(data = results, aes(y = condition, x = misreliant_payoff, fill = "Misreliant"), alp
  stat_slabinterval(data = results, aes(y = condition, x = baseline2_payoff, fill = "Baseline(human alo
  stat_slabinterval(data = results, aes(y = condition, x = baseline_payoff, fill = "Baseline"), alpha =
  stat_slabinterval(data = results, aes(y = condition, x = benchmark_payoff, fill = "Benchmark"), alpha
  # geom_vline(xintercept = as.vector(rational_benchmark)$expected_payoff, linetype = "dashed", size =
  # geom_vline(data = results, aes(xintercept = baseline_payoff, linetype = "Baseline"), size = 1) +
  labs(x = "", y = "") +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        panel.grid.major = element_line(colour = "grey"),
        axis.line.x = element_line(linewidth = 1.5, colour = "grey80"),
        panel.background = element_rect(fill = "white", color = "white"),
        axis.ticks.y = element_blank(),
        axis.ticks.x = element_line(colour = "grey")) +
  scale_fill_manual(values = colors)
```




```
# ggsave("./bansal_results.pdf", unit = "in", width = 7.22222222222, height = 3.47222222222)
```

Using discretized signals to approximate

```
rational_action = human_predictions %>%
  mutate(human_pred = as.character(human_pred)) %>%
  group_by(human_pred, pred, pred2, cluster) %>%
  mutate(human_payoff = payoff(human_pred, y),
            pred_payoff = payoff(pred, y),
            pred2_payoff = payoff(pred2, y)) %>%
  summarise(human_payoff = mean(human_payoff),
            pred_payoff = mean(pred_payoff),
            pred2_payoff = mean(pred2_payoff))
```

```
## `summarise()` has grouped output by 'human_pred', 'pred', 'pred2'. You can
## override using the `.groups` argument.
```

```
exp_data_with_reliance = exp_data %>%
  mutate(human_pred = as.character(human_pred)) %>%
  mutate(choice = as.character(choice)) %>%
  mutate(behavioral_payoff = (choice == y) * bonus) %>%
  left_join(rational_action) %>%
  # mutate(pred_better = pred,
  #        pred_better_payoff = pred_payoff) %>%
  mutate(pred_better = ifelse(pred_payoff >= pred2_payoff, pred, pred2),
         pred_better_payoff = ifelse(pred_payoff >= pred2_payoff, pred_payoff, pred2_payoff)) %>%
  group_by(drawId, drawId2, condition) %>%
  arrange(desc(pred_better_payoff - human_payoff), .by_group = TRUE) %>%
```

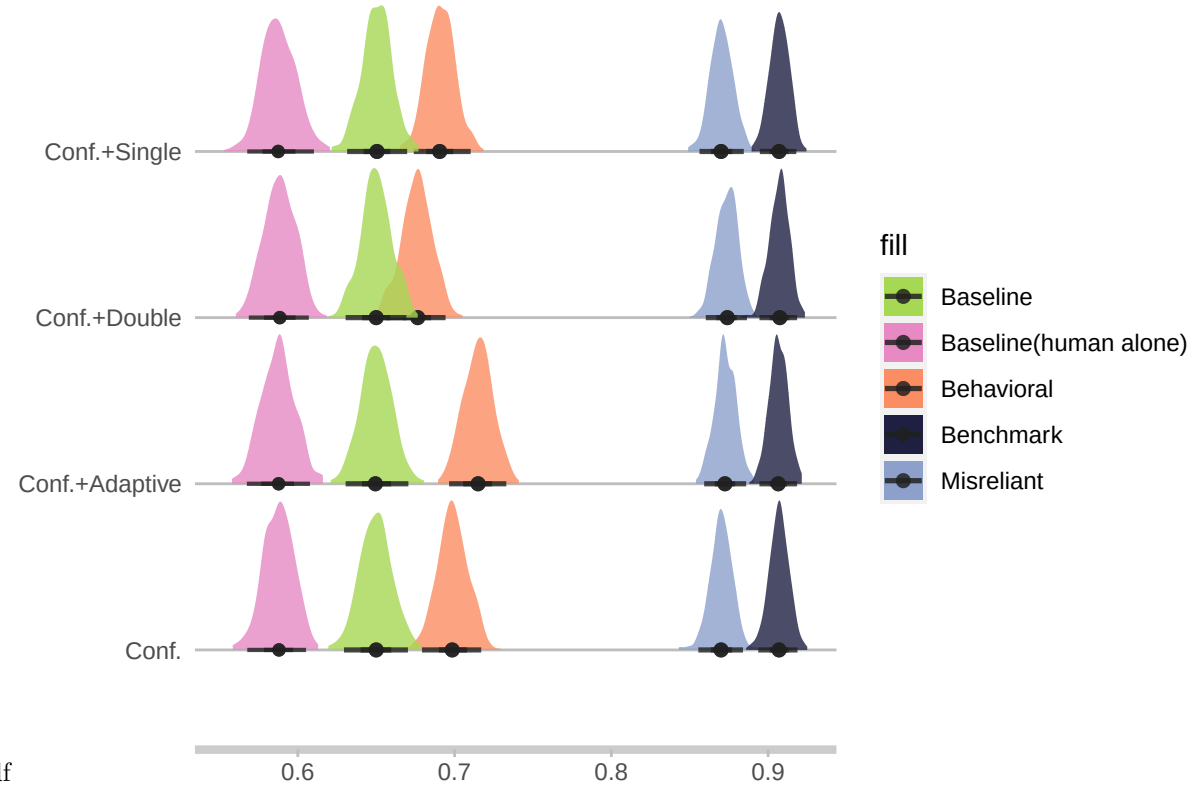
```
  mutate(sort_id = row_number()) %>%
  mutate(is_relying = ((choice == pred) | (choice == pred2)) & (choice != human_pred)) %>%
  mutate(reliance = sum(is_relying)) %>%
  ungroup() %>%
  mutate(action = ifelse(sort_id <= reliance,
                         pred_better,
                         human_pred)) %>%
  mutate(misreliant_payoff = (action == y) * bonus) %>%
  mutate(rational_benchmark_action = ifelse(pred_better_payoff > human_payoff, pred_better, human_pred)
  mutate(rational_benchmark = (rational_benchmark_action == y) * bonus) %>%
  mutate(rational_rl = ((rational_benchmark_action == pred) |
                                  (rational_benchmark_action == pred2)) &
           (rational_benchmark_action != human_pred)) %>%
  rowwise() %>%
  mutate(rational_baseline_action = pred) %>%
  mutate(rational_baseline = (rational_baseline_action == y) * bonus) %>%
  mutate(rational_baseline2_action = human_pred) %>%
  mutate(rational_baseline2 = (rational_baseline2_action == y) * bonus)
```

```
## Joining with `by = join_by(pred, pred2, human_pred, cluster)`
```

```
sample_size = 100 * 20
n_round = 500
results = data.frame()
for (i in 1:n_round) {
  results = exp_data_with_reliance %>%
    group_by(condition) %>%
    sample_n(sample_size) %>%
    # group_by(condition) %>%
    summarise(behavioral_payoff = mean(behavioral_payoff),
              misreliant_payoff = mean(misreliant_payoff),
              benchmark_payoff = mean(rational_benchmark),
              baseline_payoff = mean(rational_baseline),
              baseline2_payoff = mean(rational_baseline2),
              reliance = mean(reliance),
              rational_reliance = mean(rational_rl)) %>%
    rbind(results)
}

ggplot() +
  stat_slabinterval(data = results, aes(y = condition, x = behavioral_payoff, fill = "Behavioral"), alp
  stat_slabinterval(data = results, aes(y = condition, x = misreliant_payoff, fill = "Misreliant"), alp
  stat_slabinterval(data = results, aes(y = condition, x = baseline2_payoff, fill = "Baseline(human alo
  stat_slabinterval(data = results, aes(y = condition, x = baseline_payoff, fill = "Baseline"), alpha =
  stat_slabinterval(data = results, aes(y = condition, x = benchmark_payoff, fill = "Benchmark"), alpha
  # geom_vline(xintercept = as.vector(rational_benchmark)$expected_payoff, linetype = "dashed", size =
  # geom_vline(data = results, aes(xintercept = baseline_payoff, linetype = "Baseline"), size = 1) +
  labs(x = "", y = "") +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        panel.grid.major = element_line(colour = "grey"),
        axis.line.x = element_line(linewidth = 1.5, colour = "grey80"),
        panel.background = element_rect(fill = "white", color = "white"),
        axis.ticks.y = element_blank(),
```

```
      axis.ticks.x = element_line(colour = "grey")) +
  scale_fill_manual(values = colors)
```



signals-1.pdf

```
# ggsave("./bansal_results_test_performance.pdf", unit = "in", width = 7.22222222222, height = 3.472222
```