Supplementary Materials: Repeated CV Simulation Results

Contents

Summary

Code modifications

The following changes to the nested_cv (https://github.com/stephenbates19/nestedcv) code were made:

- Added support for weighted observations. This enabled the computation of classification performance metrics sensitive to class imbalance: recall and balanced accuracy.
- Reworked the k-fold splitting so that a stratified split is performed on the response variable.

One limitation of the nested CV method is that it requires the performance metric to be a sum of losses on individual observations. For example, classification error rate can be assessed but not AUROC.

The weighting extension enables computation of performance metrics for imbalanced data. This is essential for small molecule predictive modeling where classes are often highly imbalanced.

Nested CV intervals are tighter than standard CV intervals

To confirm that the nested method works as advertised, we checked that nested cv intervals were tighter than standard cv intervals. We found that the intervals for all performance metrics were underestimated for standard cv and were better estimated by nested CV.

Repeated 5x5 CV has comparable performance to nested CV

The nested_cv method is current state of the art for performance metric confidence interval estimation. The method is too computationally expensive for many cheminformatics applications, but we can use the intervals to perform an evaluation on representative cheminformatics data sets and then evaluate how close to optimal more practical approximations like McNemar and Dietterich's 5X2 are, and whether there are potentially better settings for repeated CV instead of 5X2.

Note that here we are estimating confidence intervals for performance of a single method, and not a difference between two methods. Statistical testing has not yet been worked out for the nested CV method, though the authors allude to future work. We needed to adapt the methods for McNemar and Dietterich's 5x2 to confidence intervals for a single metric. We use equivalent methods with the same assumptions. For McNemar, we simply obtain binomial proportion intervals.

We note that many resources on ML method comparison cite the guidance from Dietterich's paper (https://pubmed.ncbi.nlm.nih.gov/9744903/). From Raschka (https://arxiv.org/abs/1811.12808): "The bottom line is that McNemar's test is a good choice if the datasets are relatively large and/or the model fitting can only be conducted once. If the repeated fitting of the models is possible, the 5x2cv test is a good choice as it also considers the effect of varying or resampled training sets on the model fitting"

The problem with Dietterich's paper is that it was written in 1998. The guidance was derived on data sets with only 300 observations. Data sets are much larger these days. This has an impact on the correlation in performance metrics across cv folds. For a data set with a few thousand observations, one would expect less correlation in the training and test sets when iterating over CV replicates, so a larger number of replicates and CV folds may be tolerated. To test this, we implemented a repeated CV procedure, which allows for any setting of cv folds and iterations. We found that Dietterich's 5X2 interval estimates were often unstable. 5x5 CV provided a better, more stable estimate. Further optimization with more iterations of repeated CV may be possible, but we didn't want to go much larger than 5x5 = 25 fits for general guidance.

Conclusions

We found 5x5 CV CIs provided intervals with good coverage of test set performance and reasonably low expected width. These results suggest that statistical tests utilizing 5x5 CV will have good control of type I error rate and statistical power. Other advantages of 5x5 CV over nested CV are that it can be easily extended to any performance metric and methods for statistical testing based on repeated CV have already been developed and studied.

Figure 1: Interval stability comparison

A single test set split was performed. Test set performance is shown with a dashed line. 30 iterations of CI estimation were performed with the remaining data.

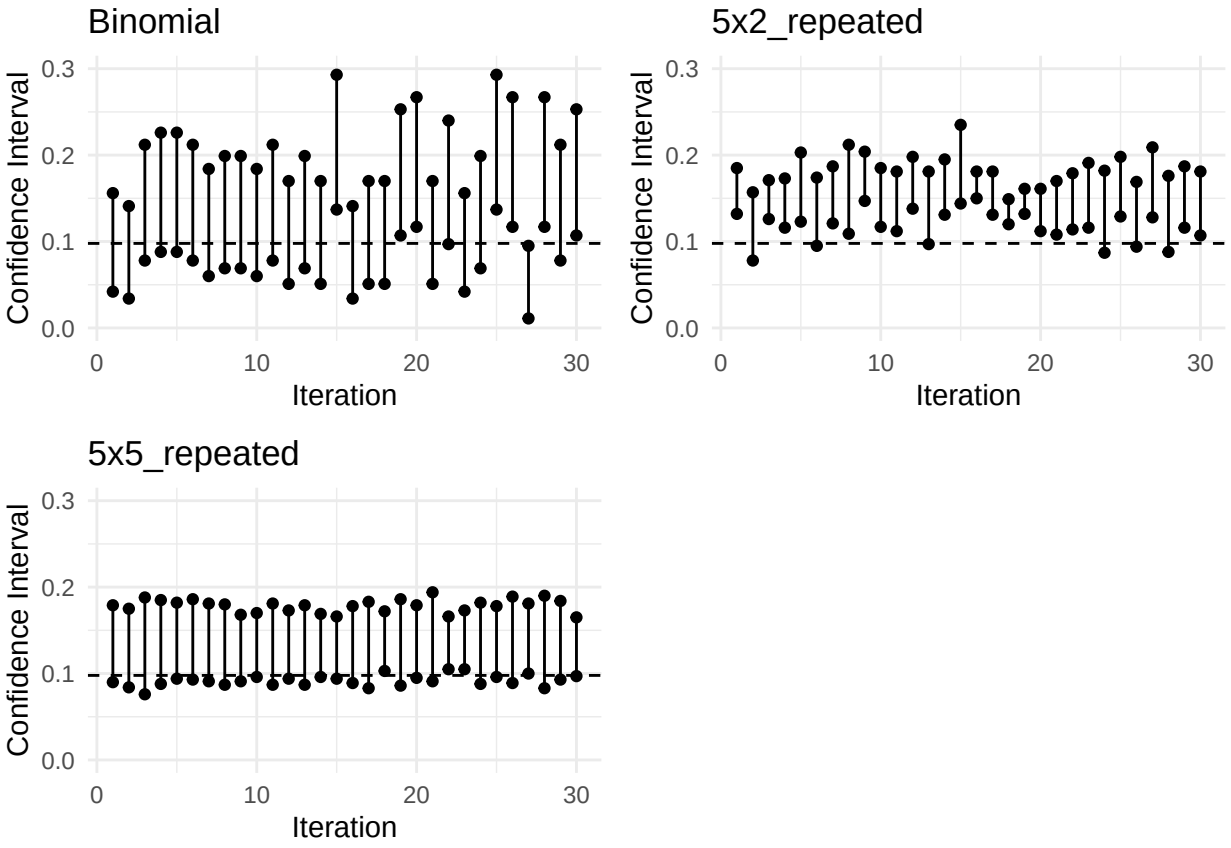
For the binomial method, intervals tend to be much too large. For Dietterich's 5x2, intervals are unstable and frequently do not cover test set performance. 5x5 repeated CV has much more stable intervals with frequent coverage of test set performance.

```
results_ls <- readRDS("results_interval_comparison.rds")
results_ls <- results_ls[1:4]

for(i in seq_along(results_ls)){
  results_ls[[i]] <- results_ls[[i]] %>%
    mutate(method = ifelse(method == "repeated_cv_dietterich", "5x5_repeated", method)) %>%
    mutate(method = ifelse(method == "dietterich_5x2", "5x2_repeated", method))
}
```

```
plot_confidence_intervals <- function(results_df, method, plot_title, ylim) {
  results_df_repeated <- results_df[results_df$method == method, ]

  ggplot(results_df_repeated, aes(x = iteration, y = ci_lo, ymax = ci_hi, ymin = ci_lo)) +
    geom_errorbar(width = 0.2) +
    geom_point(aes(y = ci_lo)) +
    geom_point(aes(y = ci_hi)) +
    geom_hline(aes(yintercept = true_error), linetype = "dashed", color = "black") +
    ylim(ylim) +
    labs(title = plot_title,
         x = "Iteration",
         y = "Confidence Interval",
         color = "Method") +
    theme_minimal()
}


plot_titles <- c("Binomial", "5x2_repeated", "5x5_repeated")
methods <- c("binom_ci", "5x2_repeated", "5x5_repeated")
plots <- list()
combined_plot <- list()
ylim_ls <- list(c(0, .3), c(0, 1), c(0, .5), c(0, .8))

for (j in seq_along(results_ls)){
  for (i in seq_along(methods)) {
    plots[[i]] <- plot_confidence_intervals(results_ls[[j]], methods[i], plot_titles[i], ylim_ls[[j]])
  }
  combined_plot[[j]] <- plot_grid(plots[[1]], plots[[2]], plots[[3]],
                                  ncol = 2, align = "hv")
}
```

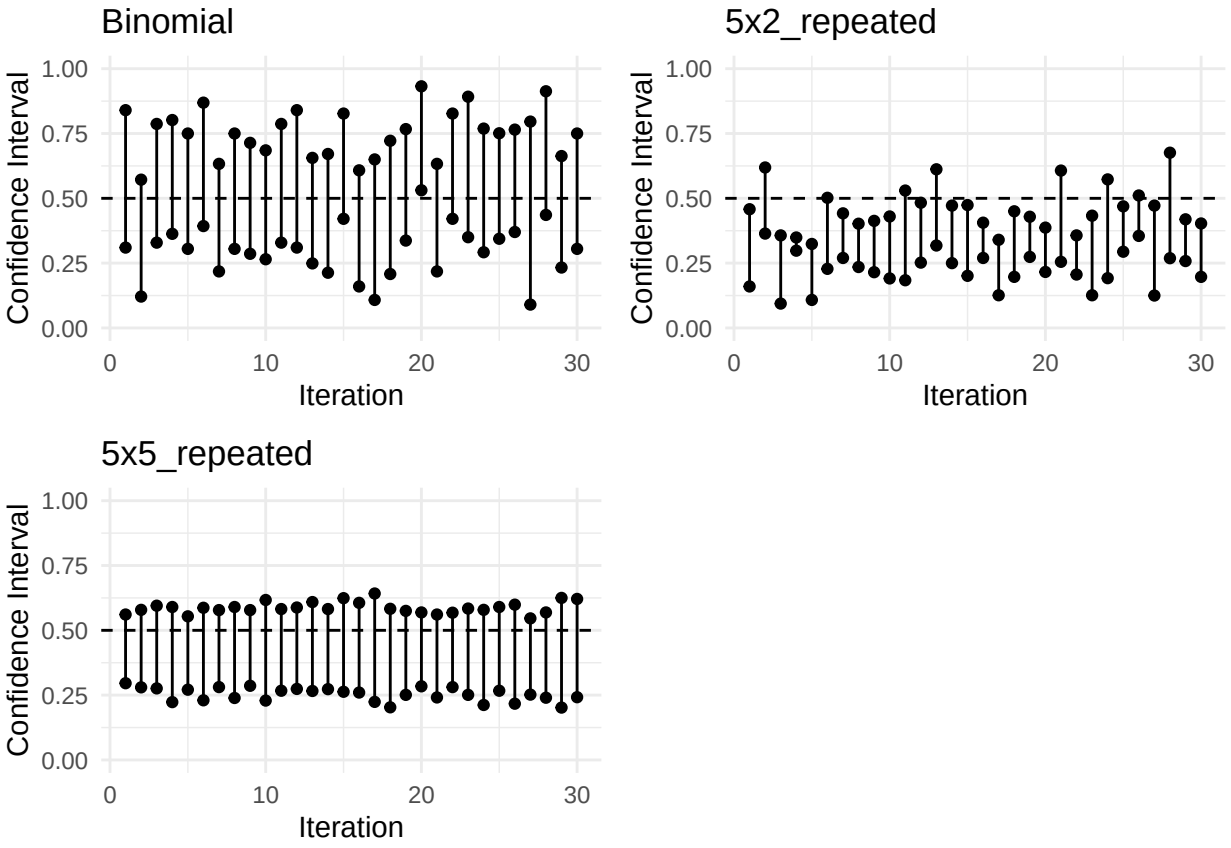
Error rate

```
print(combined_plot[[1]])
```



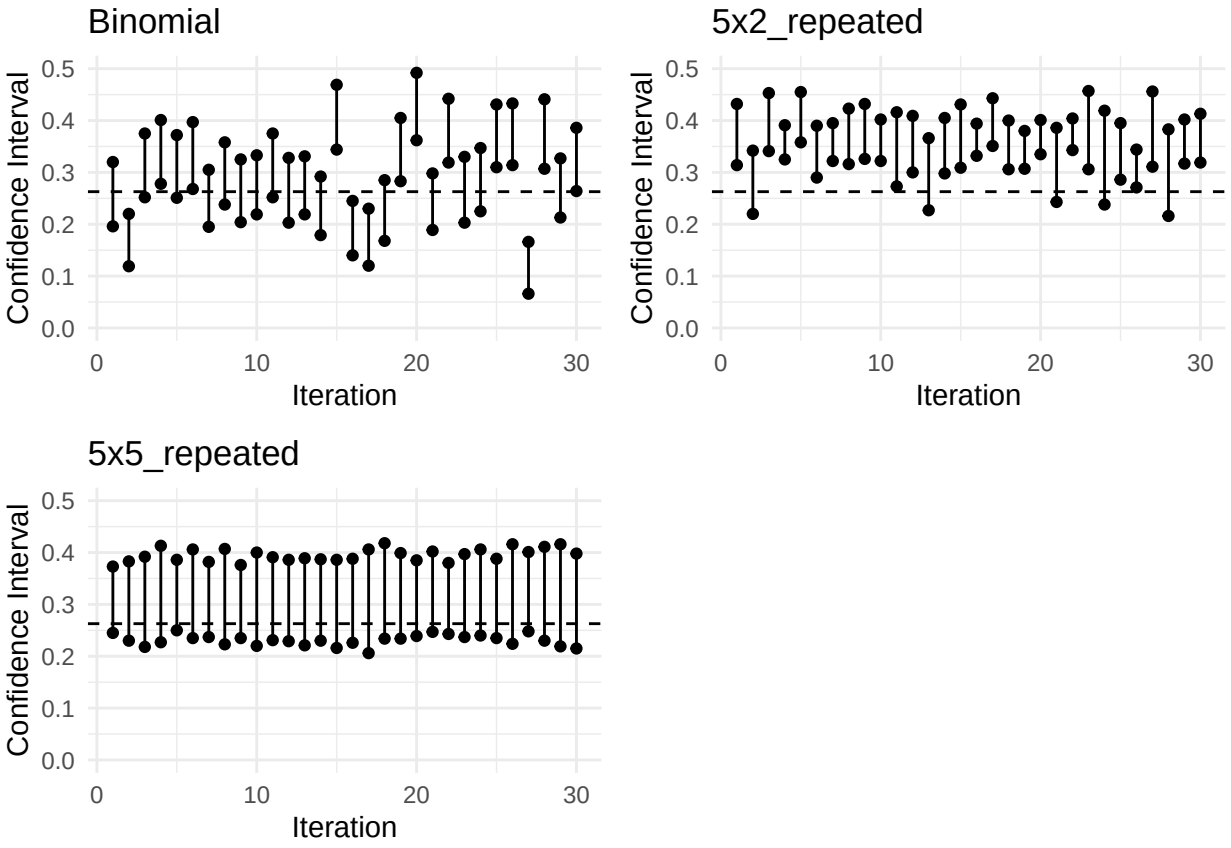
Recall

```
print(combined_plot[[2]])
```



Balanced error rate

```
print(combined_plot[[3]])
```



Entropy

```
print(combined_plot[[4]])
```

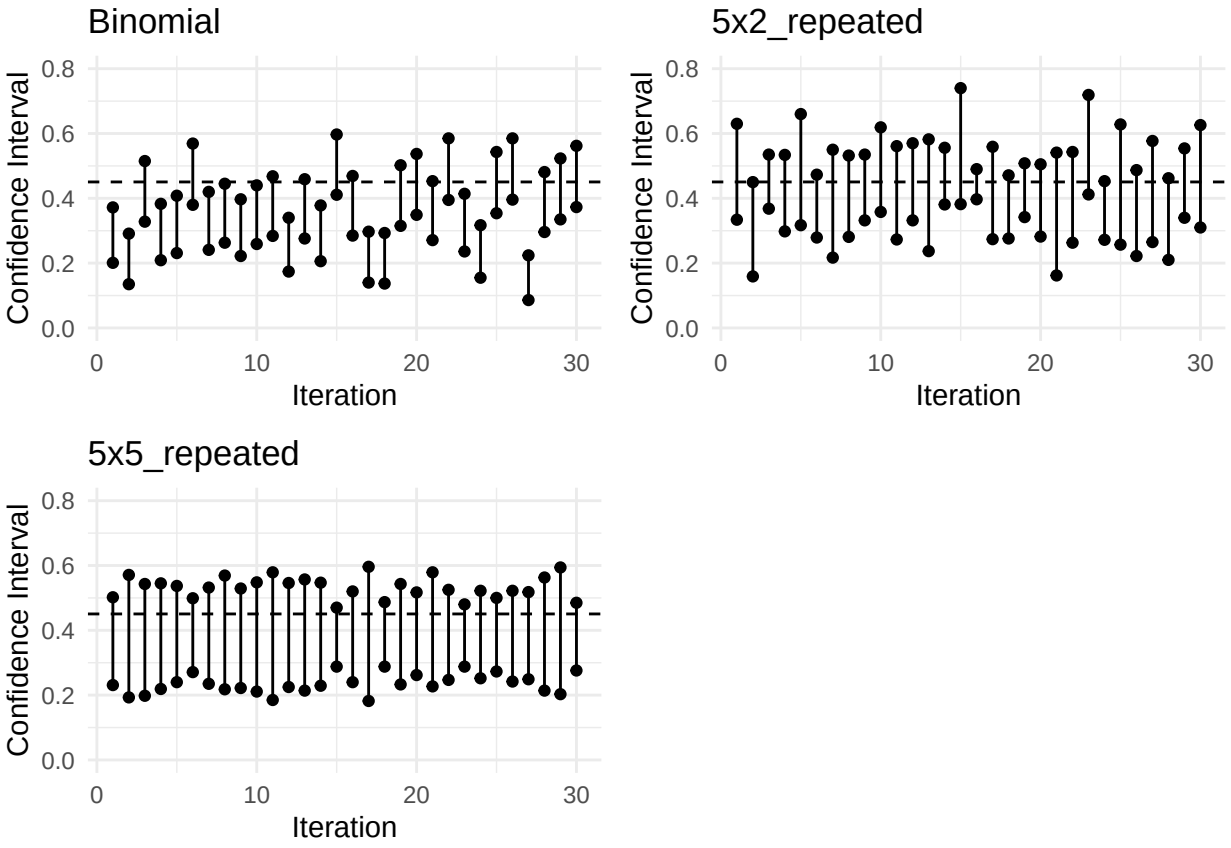


Figure 2: Interval method comparison

Comparison of CI estimation methods for one iteration. Intervals are underestimated for standard CV. Intervals are larger and cover the true test set performance for nested CV. 5x5 repeated CV also covers the true test set performance. Interval widths are either underestimated or overestimated for binomial and Deiterrich's 5x2.

```
plot_confidence_intervals <- function(results_df, plot_title) {
  results_df_method_compare <- results_df[results_df$iteration == 1, ]

  ggplot(results_df_method_compare,
         aes(x = method, y = ci_lo, ymax = ci_hi, ymin = ci_lo)) +
    geom_errorbar(width = 0.2, aes(color = method)) +
    geom_point(aes(y = ci_lo, color = method)) +
    geom_point(aes(y = ci_hi, color = method)) +
    geom_hline(aes(yintercept = true_error), linetype = "dashed", color = "black") +
    annotate("text", x = length(unique(results_df_method_compare$method)) + 0.5,
             y = unique(results_df_method_compare$true_error),
             label = "Test Error",
             hjust = 1, vjust = 1.5, color = "black", size = 3.5) +
    labs(x = "Method",
         y = "Confidence Interval",
         title = plot_title) +
    theme_minimal() +
    theme(legend.position = "none",
          axis.text.x = element_text(angle = 45, hjust = 1))
}

plot_titles <- c("Error Rate", "Recall", "Balanced Error Rate", "Cross Entropy")
plots <- list()

for (i in 1:length(results_ls)) {
  plots[[i]] <- plot_confidence_intervals(results_ls[[i]], plot_titles[i])
}

combined_plot <- plot_grid(plots[[1]], plots[[2]], plots[[3]], plots[[4]],
                           ncol = 2, align = "hv")

print(combined_plot)
```

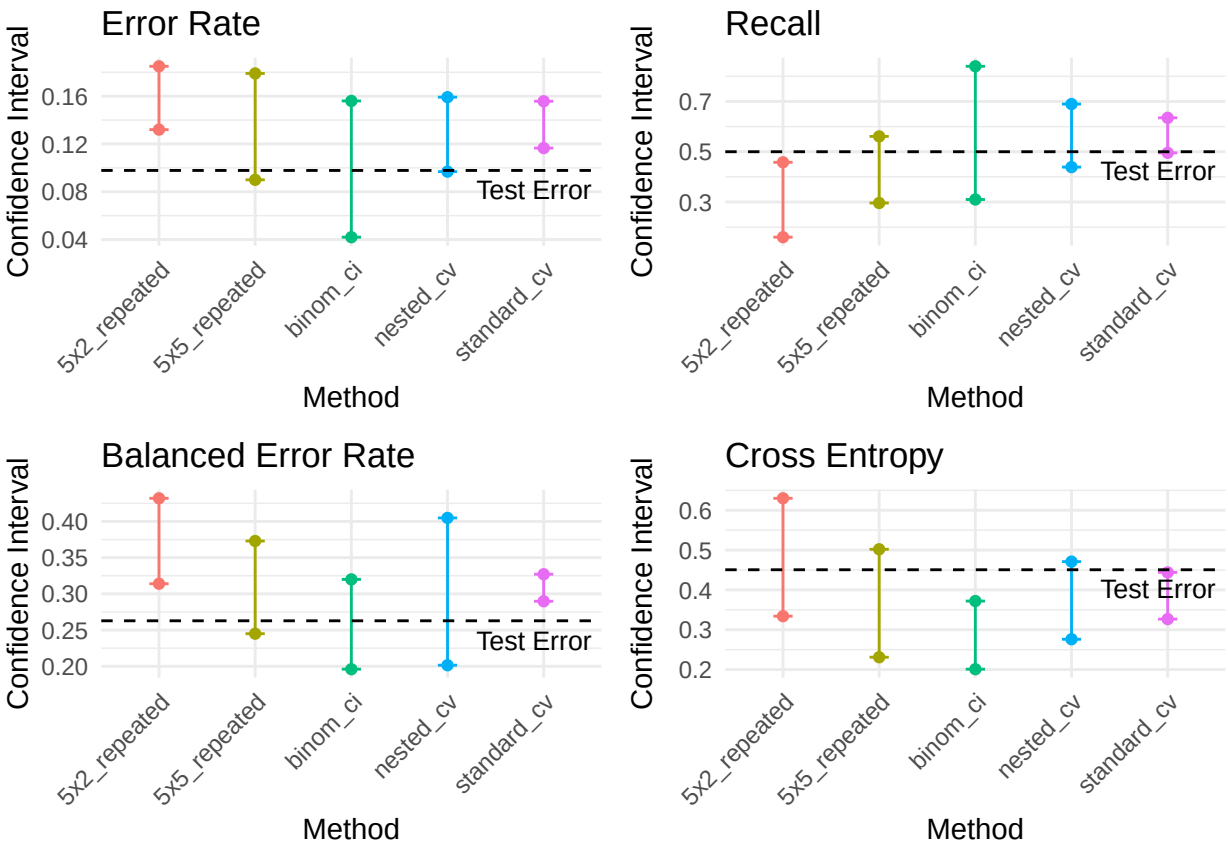

Error Rate
Confidence Interval
0.16
0.12
0.08
0.04
Test Error
5x2_repeated
5x5_repeated
binom_ci
nested_cv
standard_cv
Method
Recall
Confidence Interval
0.7
0.5
0.3
Test Error
5x2_repeated
5x5_repeated
binom_ci
nested_cv
standard_cv
Method
Balanced Error Rate
Confidence Interval
0.40
0.35
0.30
0.25
0.20
Test Error
5x2_repeated
5x5_repeated
binom_ci
nested_cv
standard_cv
Method
Cross Entropy
Confidence Interval
0.6
0.5
0.4
0.3
0.2
Test Error
5x2_repeated
5x5_repeated
binom_ci
nested_cv
standard_cv
Method

Figure 3: Coverage probability and interval width

20 iterations of test set splitting and CV estimation were performed. It took more than a day to run this simulation due the computational complexity of nested CV. Coverage probabilities and average interval widths were assessed.

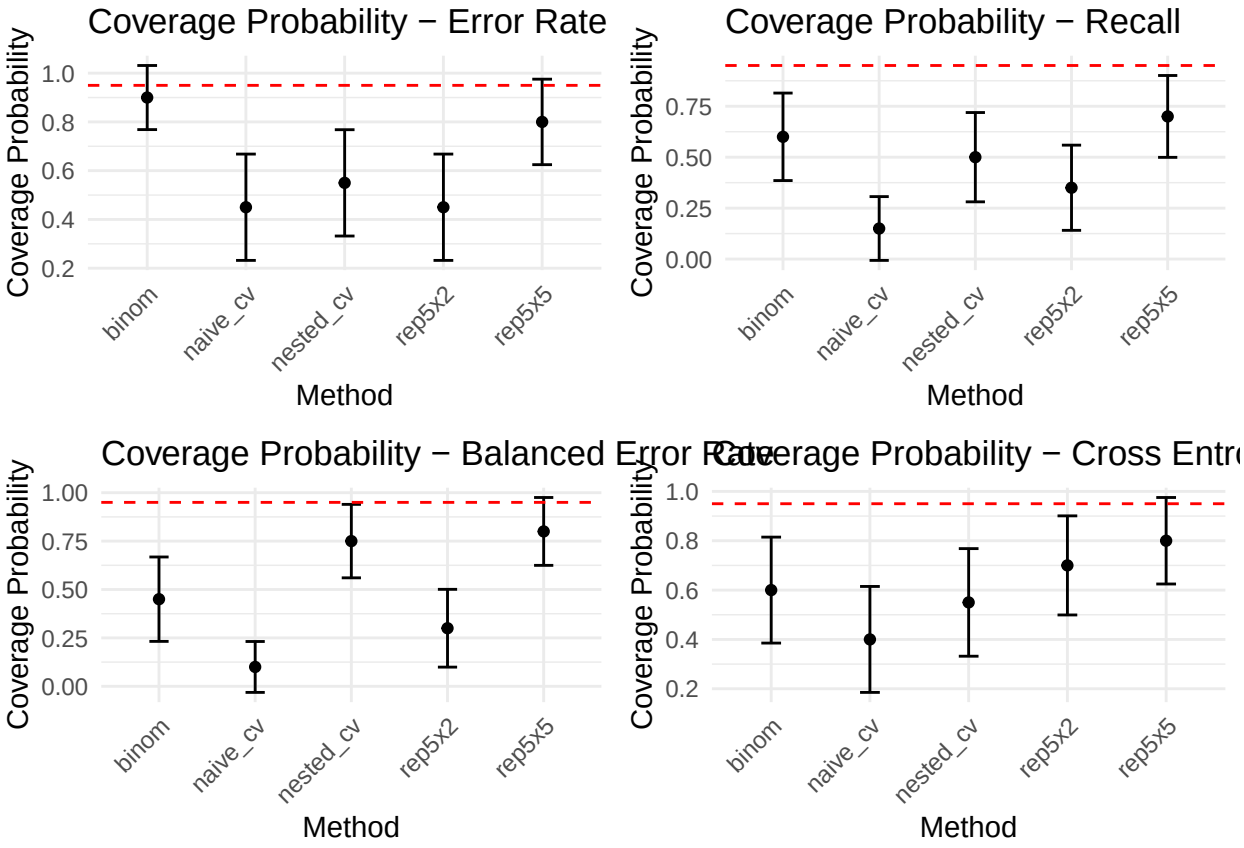
All methods are approximate CI estimation methods, so we do not expect to achieve 95% coverage exactly. Only 5x5 CV has good coverage probability consistently across metrics. Interestingly nested CV is substantially below 95% for some metrics.

```
results_ls <- readRDS("results_coverage_probability.rds")

reshape_data <- function(df) {
  df_long <- df %>%
    pivot_longer(cols = starts_with("nested_cv_ci_lo"):starts_with("rep5x5_ci_hi"),
                 names_to = c("method", ".value"),
                 names_pattern = "(.*)_ci_(.*)") %>%
    rename(ci_lo = lo, ci_hi = hi)

  return(df_long)
}

create_plots <- function(df, plot_title_suffix) {

  df <- df %>%
    mutate(interval_width = ci_hi - ci_lo,
           covers_true_error = ci_lo <= true_error & ci_hi >= true_error)

  coverage_data <- df %>%
    group_by(method) %>%
    summarise(coverage_probability = mean(covers_true_error),
              interval_width = mean(interval_width),
              coverage_se = sqrt(coverage_probability * (1 - coverage_probability) / n()),
              interval_width_se = sd(ci_hi - ci_lo) / sqrt(n()))

  coverage_plot <- ggplot(coverage_data, aes(x = method, y = coverage_probability)) +
    geom_point() +
    geom_errorbar(aes(ymin = coverage_probability - 1.96 * coverage_se,
                      ymax = coverage_probability + 1.96 * coverage_se), width = 0.2) +
    geom_hline(yintercept = 0.95, linetype = "dashed", color = "red") +
    labs(title = paste("Coverage Probability -", plot_title_suffix),
         y = "Coverage Probability", x = "Method") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))

  interval_width_plot <- ggplot(coverage_data, aes(x = method, y = interval_width)) +
    geom_point() +
    geom_errorbar(aes(ymin = interval_width - 1.96 * interval_width_se,
                      ymax = interval_width + 1.96 * interval_width_se), width = 0.2) +
    labs(title = paste("Interval Width -", plot_title_suffix),
         y = "Interval Width", x = "Method") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
  return(list(coverage_plot = coverage_plot, interval_width_plot = interval_width_plot))
}

all_plots <- list()

names(results_ls) <- c("Error Rate", "Recall", "Balanced Error Rate", "Cross Entropy")

for (i in seq_along(results_ls)) {
  plot_title_suffix <- names(results_ls)[i]
  df <- results_ls[[i]]

  df_long <- reshape_data(df)

  plot_list <- create_plots(df_long, plot_title_suffix)

  all_plots[[paste0("coverage_plot_", i)]] <- plot_list$coverage_plot
  all_plots[[paste0("interval_width_plot_", i)]] <- plot_list$interval_width_plot
}

coverage_grid <- plot_grid(all_plots$coverage_plot_1, all_plots$coverage_plot_2,
                           all_plots$coverage_plot_3, all_plots$coverage_plot_4,
                           ncol = 2)

interval_width_grid <- plot_grid(all_plots$interval_width_plot_1, all_plots$interval_width_plot_2,
                                 all_plots$interval_width_plot_3, all_plots$interval_width_plot_4,
                                 ncol = 2)
```

Coverage probability

```
print(coverage_grid)
```



Interval width

```
print(interval_width_grid)
```