Bioinformatics, 2025, **41(2)**, btaf055
https://doi.org/10.1093/bioinformatics/btaf055
Advance Access Publication Date: 3 February 2025
Applications Note

OXFORD

Data and text mining

ipd: an R package for conducting inference on predicted data

Stephen Salerno[1], Jiacheng Miao[2], Awan Afiaz[1,3], Kentaro Hoffman[4], Anna Neufeld[5], Qiongshi Lu[2,*,†], Tyler H. McCormick[4,6,*,†], Jeffrey T. Leek[1,3,*,†]

[1]Public Health Sciences, Biostatistics, Fred Hutchinson Cancer Center, Seattle, WA 98109, United States
[2]Biostatistics and Medical Informatics, University of Wisconsin-Madison, Madison, WI 53726, United States
[3]Biostatistics, University of Washington, Seattle, WA 98195, United States
[4]Statistics, University of Washington, Seattle, WA 98195, United States
[5]Mathematics and Statistics, Williams College, Williamstown, MA 01267, United States
[6]Sociology, University of Washington, Seattle, WA 98195, United States

*Corresponding authors. Qiongshi Lu, 5426 Genetics/Biotech, 425 Henry Mall, Madison, WI 53706, United States. E-mail: qlu@biostat.wisc.edu; Tyler H. McCormick, Center for Statistics and the Social Sciences, University of Washington, Box 354320, Padelford Hall, C 14, Seattle, WA 98195, United States. E-mail: tylermc@uw.edu; Jeffrey T. Leek, Fred Hutchinson Cancer Center, Mail Stop M1-B514, 1100 Fairview Ave N., Seattle, WA 98109, United States. E-mail: jtleek@fredhutch.org.

†These authors share senior authorship.

Associate Editor: Jonathan Wren

Abstract

Summary: `ipd` is an open-source `R` software package for the downstream modeling of an outcome and its associated features where a potentially sizable portion of the outcome data has been imputed by an artificial intelligence or machine learning prediction algorithm. The package implements several recent proposed methods for inference on predicted data with a single, user-friendly wrapper function, `ipd`. The package also provides custom `print`, `summary`, `tidy`, `glance`, and `augment` methods to facilitate easy model inspection. This document introduces the `ipd` software package and provides a demonstration of its basic usage.

Availability: `ipd` is freely available on CRAN or as a developer version at our GitHub page: github.com/ipd-tools/ipd. Full documentation, including detailed instructions and a usage 'vignette' are available at github.com/ipd-tools/ipd.

1 Introduction

With the rapid advancement of artificial intelligence and machine learning (AI/ML) algorithms, and owing to financial and domain-specific constraints, researchers from a wide range of disciplines increasingly use predictions from pre-trained algorithms as outcome variables in statistical analyses (Hoffman *et al.* 2024). However, reifying algorithmically derived values as measured outcomes may lead to potentially biased estimates and anti-conservative inference (e.g. see, Wang *et al.* 2020). In particular, the statistical challenges encountered when drawing *inference on predicted data* (IPD) include: (i) understanding the relationship between the predicted outcomes and their true, unobserved counterparts, (ii) quantifying the robustness of the AI/ML models to resampling or uncertainty about the data they were trained on, and (iii) appropriately propagating both bias and uncertainty from upstream predictive model into downstream inferential procedures. We refer to methods developed to address these challenges as methods for conducting IPD.

Several recent methods have been proposed for conducting IPD. These include *post-prediction inference (PostPI)* by Wang *et al.* (2020), *prediction-powered inference (PPI)*, and *PPI++* by Angelopoulos *et al.* (2023a,b), and *post-prediction adaptive inference (PSPA)*, as well as PSPA's extensions, *POP-TOOLS* and *post-prediction sumstats-based inference (PSPS)* by Miao *et al.* (2023, 2024) and Miao and Lu (2024), respectively, *prediction-powered bootstrap (PPBoot)* by Zrnic (2024), and semi-supervised methods, *cross-prediction-powered inference (Cross-PPI)* by Zrnic and Candès (2024) and *design-based supervised learning (DSL)* by Egami *et al.* (2023). In general, these methods employ one of two strategies for IPD correction. They either (i) construct pseudo-outcomes designed to resemble the true, unobserved outcome and correct inference on the outcome space, or (ii) calibrate the parameter estimates and standard errors directly.

These methods have been developed in quick succession in response to the ever-growing practice of using predicted data directly to conduct statistical inference. To enable researchers and practitioners interested in fully using these state-of-the-art methods, we have developed `ipd`, a comprehensive open source software package that implements these existing methods under the umbrella of IPD. Moreover, we provide a convenient wrapper and helper functions, so these methods can be compared in numerous inferential settings. This note provides an overview of the package, including installation instructions, basic usage examples, and additional

Received: 14 October 2024; **Revised:** 7 January 2025; **Editorial Decision:** 27 January 2025; **Accepted:** 30 January 2025
© The Author(s) 2025. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

documentation. The examples presented here show how to generate data, fit models, and use custom methods provided by the package.

```
#-- Install the ipd package and load
install.packages("ipd")
library(ipd)
```

2 Installation and usage

`ipd` is implemented in the `R` statistical computing language (R Core Team 2024). Full documentation, including detailed downloading and installation instructions and usage 'vignettes' are available on the package website: github.com/ipd-tools/ipd. The `ipd` package has been successfully tested and runs on all the latest versions of Windows, Mac OS X, Ubuntu (Linux) operating systems. To install the development version of `ipd` from GitHub, one can use the `devtools` package.

We provide a simple example to demonstrate the basic use of this function. Following the notation of Miao *et al.* (2023), we assume the user has a dataset, $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$, consisting of $\mathcal{L} = \{(Y_i, f_i, \boldsymbol{X}_i, Set_i); i = 1, \ldots, n\}$ *labeled* samples of the observed outcome, Y, predicted outcome, f, and features of interest, $\boldsymbol{X}$, and $\mathcal{U} = \{(f_i, \boldsymbol{X}_i, Set_i); i = n+1, \ldots, n+N\}$ *unlabeled* samples, where the outcome is not observed. Set is a variable indicating whether the ith observation is *labeled*, where $Set_i = 1$ if the observation is *labeled* and $Set_i = 0$ otherwise.

The main function, `ipd`, gives access to the various methods for conducting IPD with different potential estimands. The user supplies a `formula` of the form `Y~f = X1 + X2 + ...`, where `Y` is replaced by the name of the observed outcome variable in the data, `f` is replaced by the name of the predicted outcome, and `X1`, `X2`, ..., are the names of the independent variables (features) of interest. The user has the option to supply either a stacked dataset and the name of the column corresponding to the Set label indicator, via the `data` and `label` arguments, respectively, or to supply labeled and unlabeled data sets separately, via the `data` and `unlabeled_data` arguments, respectively. Not all columns in `data` or `unlabeled_data` may be used unless explicitly referenced in the `formula` argument or in the `label` argument. Options for the available methods, specified via the `method` argument, include `"postpi_analytic"` or `"postpi_boot"` (Wang *et al.* 2020), `"ppi"` (Angelopoulos *et al.* 2023a), `"ppi_plusplus"` (Angelopoulos *et al.* 2023b), and `"pspa"` (Miao *et al.* 2023). Current estimands, specified via the `model` argument, include the population mean (`"mean"`) or qth quantile (`"quantile"`, with additional argument `q=q`) of the outcome, linear (`"ols"`), and logistic (`"logistic"`) regression. Future development will include a broader class of exponential family model parameters including multiclass logistic regression, time-to-event models, and causal targets such as average treatment effects (ATE). Additional (optional) arguments include the significance level, `alpha`, for the $100(1-\alpha)\%$ confidence intervals, and other method-specific arguments.

The function outputs a "glm"-style list with the parameter estimates, standard errors, and confidence limits, as well as additional information about the function call and intermediate estimated quantities [e.g. the estimated relationship model of Wang *et al.* (2020) or the estimated tuning parameters of Angelopoulos *et al.* (2023a,b) and Miao *et al.* (2023)].

All method-specific and helper functions are documented and exported by the `ipd` package for additional user flexibility. These include a function to generate simulated data, `simdat`, to facilitate exploration of the methods in the absence of real data, as well as `print`, `summary`, `tidy`, `glance`, and `augment` methods to facilitate easy model inspection (Wickham *et al.* 2023). Documentation for the package and these functions can be accessed by running `?ipd-package`, `?<function>` (e.g., `?simdat`) or `help("simdat")` in the R console. In the next section, we provide an illustrative example using simulated data generated by `simdat` for linear regression.

3 An example analysis

We simulate a continuous outcome for linear regression, with $Y = \beta_1 X_1 + \frac{1}{2}X_2^2 + \frac{1}{3}X_3^3 + \frac{1}{4}X_4^2 + \varepsilon$, where X_1, X_2, X_3, and $X_4 \sim \mathcal{N}(0,1)$, $\beta_1 = 1$, corresponding to the true value of the linear regression coefficient for X_1 (our target of inference), and $\varepsilon \sim \mathcal{N}(0, \sigma_Y^2); \sigma_Y = 4$ using the `simdat` function. We generate a stacked dataset of 100 training, 100 labeled, and 1000 unlabeled observations, where the corresponding set is denoted by a column called "set_label." We generate predicted outcomes by training a generalized additive model on the training set and making predictions for the labeled and unlabeled sets.

```
#-- Generate example data for linear regression
dat <- simdat(n = c(100, 100, 1000), effect = 1,
    sigma_Y = 4, model = "ols")
```

For each method, we calculate the point estimate and corresponding $100(1-\alpha)\%$ confidence interval for β_1, where $\alpha = 0.05$ and we have a two-sided hypothesis test. We compare the methods in the package to three additional models, which serve as performance benchmarks: the "oracle" regression, which fits the ideal model on the true, unknown outcome for the unlabeled observations (possible on simulated data), the "naive" regression, which treats the predicted outcomes as if they were the true, unobserved outcomes, and the "classic" regression, which utilizes only the labeled subset of the data:

```
dat_ols  <- dat[dat$set_label != "training",]
dat_ols_l <- dat_ols[dat_ols$set_label == "labeled",]
dat_ols_u <- dat_ols[dat_ols$set_label == "unlabeled",]
#-- Benchmark Regressions
fit0 <- lm(Y ~ X1, data = dat_ols_u) #- Oracle
fit1 <- lm(f ~ X1, data = dat_ols_u) #- Naive
fit2 <- lm(Y ~ X1, data = dat_ols_l) #- Classic
#-- PostPI
fit3 <- ipd(Y - f ~ X1, method = "postpi_boot", model =
      "ols", data = dat_ols, label = "set_label")
#-- PPI
fit4 <- ipd(Y - f ~ X1, method = "ppi", model = "ols", data
      = dat_ols, label = "set_label")
#-- PPI++
fit5 <- ipd(Y - f ~ X1, method = "ppi_plusplus", model =
      "ols", data = dat_ols, label = "set_label")
#-- PSPA
fit6 <- ipd(Y - f ~ X1, method = "pspa", model = "ols", data
      = dat_ols, label = "set_label")
```

As a benchmark, the hypothetical 'oracle' regression would have the correct estimate and correctly-sized confidence interval if the outcome was measured for each observation, while

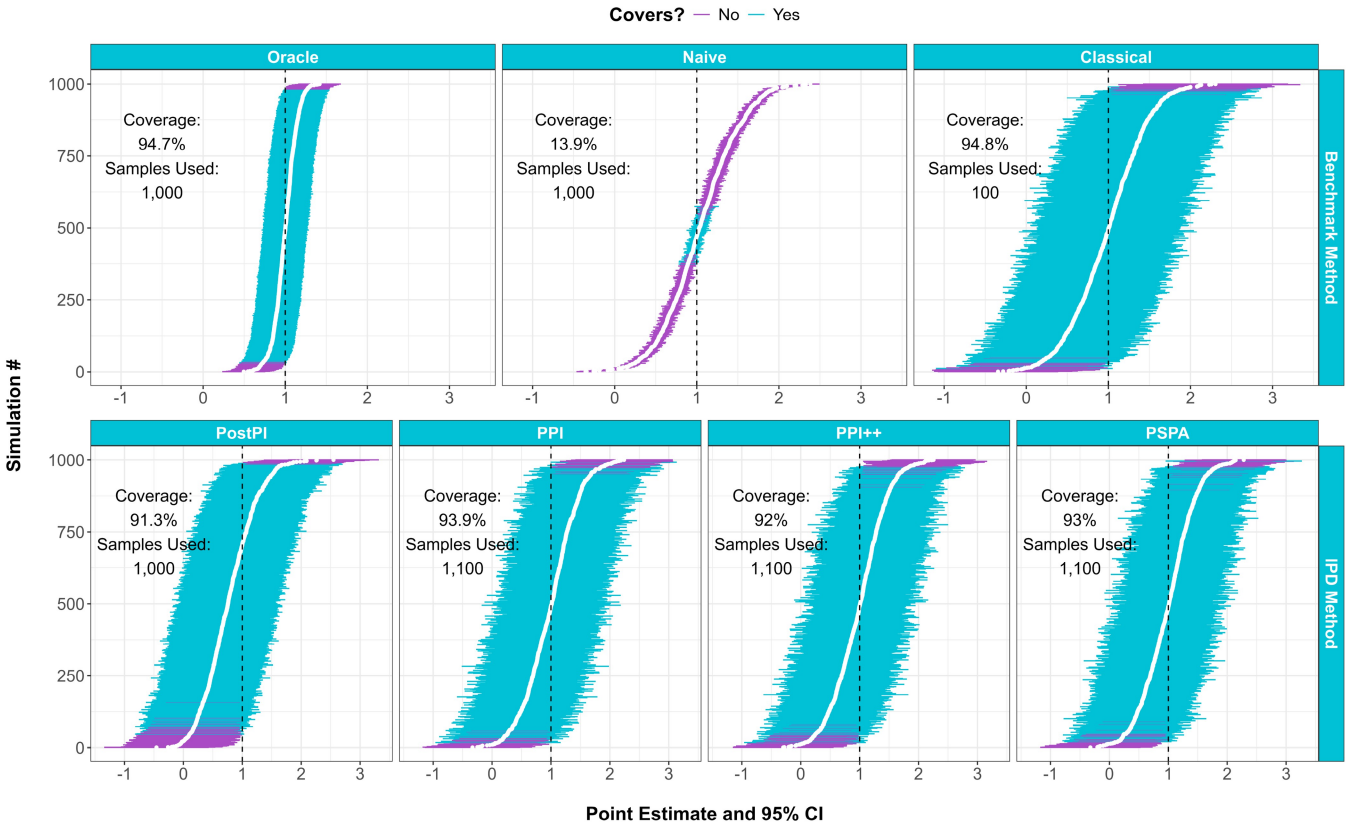


Figure 1. Point estimate and corresponding 95% confidence intervals for four available IPD methods (postpi, ppi, ppi_plusplus, and pspa; second row), as compared to three benchmark regressions (oracle, naive, and classical; first row) on 1000 simulated linear regression datasets.

in practice, the "naive" regression may have a biased point estimate and a confidence interval that is too narrow. The "classical" regression will have the correct estimate but wider confidence interval, as it only uses the labeled subset of the data. The implemented IPD methods all correctly estimate the coefficient and have confidence intervals that are wider than the "oracle" but narrower than the "classic" regression (Fig. 1).

4 Printing, summarizing, and tidying

The package also provides custom `print`, `summary`, `tidy`, `glance`, and `augment` methods to facilitate easy model inspection. Namely, the `print` method gives an abbreviated summary of the output from the `ipd` function, the `summary` method gives more detailed information about the estimated coefficients, standard errors, and confidence limits, The `tidy` method organizes the model coefficients into a "tidy" format (Wickham *et al.* 2023), the `glance` method returns a one-row summary of the model fit, and the `augment` method adds model predictions and residuals to the original dataset. For a more detailed look into using the `ipd` wrapper function and the method-specific individual functions, please refer to the vignettes provided with the package.

```
print(fit3)                          #- Print
summ <- summary(fit3)                #- Summary
print(summ)                          #- Print Summary
tidy(fit3)                           #- Tidy
glance(fit3)                         #- Glance
augmented_df <- augment(fit3)        #- Augment
```

5 Conclusion

In this note, we present `ipd`, a comprehensive `R` package which implements various recent methods for conducting inference on predicted data. We highlight the usability of this software for practitioners to draw valid inference on algorithmically derived outcomes, as well as the ability to facilitate comparisons for those wishing to further develop methods in the space of IPD. It is our hope that we, and others members of the research community, will maintain and grow this package as the field of IPD continues to mature in the current AI/ML era.

Conflict of interest: J.T.L. reports teaching Coursera courses which generate revenue for both Johns Hopkins University and the Fred Hutchinson Cancer Center. J.T.L. reports co-founding and serving on the board of Synthesize Bio.

Funding

This work was supported in part by the National Institutes of Health, National Institute of General Medical Sciences, Grant No. R35 GM144128 and the Fred Hutchinson Cancer Center, J. Orin Edson Foundation Endowed Chair (S.S. and J.T.L.), the National Institutes of Health, National Human Genome Research Institute, Grant No. U01 HG012039 (J.M. and Q.L.), and the National Institutes of Health, National Institute of Mental Health, Grant Nos. DP2 MH122405, R01 HD107015, and P2C HD042828 (T.H.M.).

Data availability

No new data were generated or analysed in support of this research.

References

Angelopoulos AN, Bates S, Fannjiang C *et al*. Prediction-powered inference. *Science* 2023a;**382**:669–74.

Angelopoulos AN, Duchi JC, Zrnic T. PPI++: Efficient prediction-powered inference. arXiv, arXiv:2311.01453, 2023b, preprint: not peer reviewed.

Egami N, Hinck M, Stewart B *et al*. Using imperfect surrogates for downstream inference: design-based supervised learning for social science applications of large language models. *Adv Neural Inf Process Syst* 2023; **36**:68589–601.

Hoffman K, Salerno S, Afiaz A *et al*. Do we really even need data? arXiv, arXiv:2401.08702, 2024, preprint: not peer reviewed.

Miao J, Lu Q. Task-agnostic machine learning-assisted inference. *Adv Neural Inf Process Syst* 2024;**37**:106162–89.

Miao J, Miao X, Wu Y *et al*. Assumption-lean and data-adaptive post-prediction inference. arXiv, arXiv:2311.14220, 2023, preprint: not peer reviewed.

Miao J, Wu Y, Sun Z *et al*. Valid inference for machine learning-assisted genome-wide association studies. *Nat Genet* 2024;**56**:2361–9.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2024.

Wang S, McCormick TH, Leek JT. Methods for correcting inference based on outcomes predicted by machine learning. *Proc Natl Acad Sci USA* 2020;**117**:30266–75.

Wickham H, Vaughan D, Girlich M. tidyr: Tidy Messy Data, 2023. R package version 1.3.0.

Zrnic T. A note on the prediction-powered bootstrap. arXiv, arXiv:2405.18379, 2024, preprint: not peer reviewed.

Zrnic T, Candès EJ. Cross-prediction-powered inference. *Proc Natl Acad Sci USA* 2024;**121**:e2322083121.

© The Author(s) 2025. Published by Oxford University Press.

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.
Bioinformatics, 2025, 41, 1–4
https://doi.org/10.1093/bioinformatics/btaf055
Applications Note