





You are what you click: using machine learning to model trace data for psychometric measurement

Richard N. Landers^a , Elena M. Auer^a , Gabriel Mersy^{a,b},
Sebastian Marin^a and Jason Blaik^c

^aDepartment of Psychology, University of Minnesota, Minneapolis, MN, USA; ^bDepartment of Computer Science, University of Chicago, IL, USA; ^cCappfinity, Sydney, Australia

ABSTRACT

Assessment trace data, such as mouse positions and their timing, offer interesting and provocative reflections of individual differences yet are currently underutilized by testing professionals. In this article, we present a 10-step procedure to maximize the probability that a trace data modeling project will be successful: 1) grounding the project in psychometric theory, 2) building technical infrastructure to collect trace data, 3) designing a useful developmental validation study, 4) using a holdout validation approach with collected data, 5) using exploratory analysis to conduct meaningful feature engineering, 6) identifying useful machine learning algorithms to predict a thoughtfully chosen criterion, 7) engineering a machine learning model with meaningful internal cross-validation and hyperparameter selection, 8) conducting model diagnostics to assess if the resulting model is overfitted, underfitted, or within acceptable tolerance, and 9) testing the success of the final model in meeting conceptual, technical, and psychometric goals. If deemed successful, trace data model predictions could then be engineered into decision-making systems. We present this framework within the broader view of psychometrics, exploring the challenges of developing psychometrically valid models using such complex data with much weaker trait signals than assessment developers have typically attempted to model.

KEYWORDS

Machine learning;
trace data; data science;
mousetrap; psychometric

CONTACT Richard N. Landers  rlanders@umn.edu  Department of Psychology, University of Minnesota, Minneapolis, MN, USA.

Copyright © 2022 The Author(s). Published with license by Taylor & Francis Group, LLC.

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License (<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited, and is not altered, transformed, or built upon in any way.

As digital assessment technologies grow more complex, so do the trace data that they produce and the potential of those trace data to be repurposed for additional valid assessment. The term, *trace data*, sometimes called *digital traces*, *digital footprints*, *digital exhaust* or *behavioral surplus*, refers to usually-incidental behavioral metadata created and recorded in the wake of other digital activities. Although the science of repurposing such data is in early stages, it capitalizes on two specific technological advances. First, the devices that people typically use to complete remote digital assessments, such as smartphones with mobile broadband, now have such powerful data processing capabilities and high data transfer speeds that the additional burden of collecting, preprocessing, and streaming trace data back to assessment servers has become trivial throughout the developed world. Second, although the sheer volume and variety of such data would have even ten years ago been difficult to meaningfully process, recent advances in machine learning have made the development of predictive models using such data within reach of even modestly priced personal computers. Combining an ever-growing variety of potentially useful data with advanced analytic techniques optimized to the nature of such data offers the potential for meaningful psychological measurement in situations that were previously infeasible or impossible.

Despite its promise, working at this frontier demands that researchers and developers walk a line between complex measurement concepts and data analytic approaches that are each generally unfamiliar to those on the opposite side of the line. Whereas assessment experts need to develop new skills in data wrangling, data architectures, and machine learning, data science experts need to develop their knowledge of psychometric concepts, like reliability and validity, and traditional psychometric test development procedures. It is only with both skill sets that trace data can be meaningfully analyzed for construct measurement. Without data science, the data are far too complex for traditional analytic approaches. Without psychometrics, it is quite easy to develop models that appear technically sufficient during development yet fail in real-world application scenarios. Similarly, it is difficult to make claims about the psychological constructs likely being assessed in trace data models without a foundation in psychometric theory. Knowing which constructs are being measured is generally an important criterion for the success of a psychometric assessment.

In the present article, we present a framework for the development of trace data-based assessments of known constructs, including both theoretical and technical dimensions. Importantly, because trace data vary so dramatically in their specific nature across technical applications, there is no single checklist procedure that can always be followed to create useful trace data models. For example, there are myriad technical

differences between the modeling of traces from a job applicant's interactions with an employment website and the modeling of mid- assessment traces created by a job applicant playing a game-based assessment. Given this diversity of potential application, we present an overall model engineering process, shown in [Table 1](#), that can be followed regardless of dataset-specific roadblocks, highlighting where common issues are likely to be experienced and providing advice on how to navigate them. In doing so, we hope to encourage the creation of more psychometrically meaningful trace data models, offering expanded value for the creators of digital assessments at relatively little cost.

To illustrate the complex decision-making that occurs at each of these steps more concretely, we focus on the analysis of spatiotemporal interaction data, a common type of trace data across many application domains. In the case of desktop and laptop use, spatiotemporal interactions generally focus on mouse movement, capturing locations and associated times of both the mouse cursor and mouse clicks. In the case of mobile devices, the locations and times of finger taps are commonly recorded. Although the term *trace data* can refer to any metadata created as a byproduct of some other digital activity, analysis of spatiotemporal interaction data is perhaps the most common trace data analysis due to the ubiquity of such data, which are generated and can be recorded by virtually any digital application. An example of spatiotemporal trace data analysis for the measurement of constructs can be found in the work of Auer et al. (2022), who conducted secondary analysis of trace data created by a game-based assessment of general cognitive ability validated by Landers et al. (2022). Although Auer et al. were unable to predict general cognitive ability at a sufficiently high level to support measurement equivalence ($R^2 = .434$), they were able to predict a criterion measure slightly better with the trace data model than with general cognitive ability measure. This suggested either improved measurement of cognitive ability or the reflection of additional constructs beyond it by the cognitive ability assessment's traces. This potential, in combination with the low cost of data collection, makes spatiotemporal interaction data analysis a common and provocative starting point for those seeking to find value in digital assessments beyond their originally intended psychometric purposes. It also provides an accessible entry point to trace data analysis more broadly, using more diverse types of data.

A framework for measuring constructs with trace data

Although we present trace data modeling in [Table 1](#) as a series of steps, like many modeling techniques associated with machine learning, it should be approached as an iterative engineering process (Amershi et al.,

Table 1. Steps to develop meaningful predictive trace data models.

Step	Name	Description	Example
1	Theoretical Grounding	Identify constructs potentially reflected by trace data given existing measurement or psychometric theory. Set and specify the psychometric standards that will be used to evaluate the final developed scores.	Cognitive ability is reflected in cognitively complex tasks, and mouse movements may reflect cognitive processes. Specify threshold validity needed to be .7 to reflect adequate predictive accuracy.
2	Trace Data Architecture	Identify and build infrastructure to record trace data that can be linked to identified constructs.	Augment an existing assessment to collect mouse movement data at a desirable resolution given technical constraints (e.g., 60 Hz, i.e., 60 measurements per second).
3	Development Study Design	Conduct a research study with both traditional measurement of target construct(s) and newly designed trace data system.	Administer the original assessment with trace data capture plus a questionnaire measure (or measures) of cognitive ability.
4	Training/Test Split	Before any dataset manipulation other than listwise deletion for overtly problematic cases, split the development study dataset into training and test sets.	Using R or Python, randomly assign 80% of cases to training and 20% to test.
5	Data Wrangling: Pre-processing, Exploratory Analysis, and Feature Engineering	Using the development training set, develop meaningful variables/features as appropriate; also consider imputation for missing data at this stage.	Convert raw mouse coordinates into variables capturing gross and precise movement, as appropriate to the assessment, using established R or Python libraries, or novel code.
6	Algorithm and Hyperparameter Identification	Depending upon the specific prediction problem (i.e., nature of the criterion, predictor set, and sample size), identify a set of appropriate machine learning algorithms and their hyperparameters for testing.	Determine elastic net, support vector machine, and extreme gradient boosted trees as appropriate and feasible given dataset characteristics; given no a priori expectations about relevant hyperparameters, select a distribution of each evenly spaced along their ranges for testing.
7	Model Fitting and k-Fold Cross- Validation	Fit the model using k-fold cross-validation with identified hyperparameters to determine best- performing model.	With 3 algorithms, each with 3 hyperparameters of interest across 10 folds, fit all 900 models; use mean and variance of cross- validated R2 values to identify best-performing algorithm and hyperparameter settings.
8	Model Refinement and Revision	Conduct model diagnostics to determine if any model revision would be appropriate (requiring returning at any stage prior to this one); do not reference the test set for any decision-making.	In diagnostics, observe that an entire class of mouse movement features do not contribute; consider revising approach from Step 5 or simply dropping this set of predictors and rerunning from Step 7.

(Continued)

Table 1. (Continued).

Step	Name	Description	Example
9	Holdout Validation and Psychometric Evaluation	Using the final model, pre-process the test set as the training set was pre-processed, fit values in the test set, and observe if accuracy is similar or greater than during k-fold. If not, model likely capitalizes upon artifacts in the training set and caution is warranted. Assess if model meets psychometric standards set in Step 1.	Fit cross-validated model to test set, observing that the mean k-fold cross-validation R2 is slightly smaller than the held out test-set R2; accept the final model as appropriate. If the holdout R2 was much lower, develop a new test set and repeat from Step 4, pursuing a less complex model to improve generalizability. Calculate final validity and assess if it met specified standard. If yes, proceed. If no, trace data model should not be used.

2019; Landers & Marin, 2021). Successfully engineering a predictive trace data model that can meaningfully predict criteria in new, unseen data may require revisiting earlier steps to alter key decisions and try again, adding or repeating steps as new requirements or limitations are discovered, or other such procedural modifications. This approach, given its emphasis on revising analytic processes after observing results, can mimic what in other circumstances assessment experts might consider questionable research practices, such as hypothesizing after results are known (Murphy & Aguinis, 2019). However, there are several analytic techniques to reduce the risk of overfitting despite these practices, which we will highlight where relevant. Regardless, these steps should be considered guidelines rather than prescriptive of a high-quality engineering process. The engineering requirements of any particular project may have unique components, so progression through these steps should be considered a minimum rather than comprehensive. As commonly employed in systems engineering (Landers et al., 2022), iteration through these steps, as needed, is recommended.

Step 1: Theoretical grounding

Trace data analysis can be conceptually challenging to assessment professionals because of the nature of the data being analyzed. A single trace datum is often difficult or impossible to interpret psychologically. Whereas a single question on a questionnaire is generally designed to strongly correlate with a targeted latent trait, such that random error variance across multiple questions in a finalized measure can be reasonably assumed to average to zero, item-trait correlations are very weak in trace data analysis, and traditional classical test theory assumptions

often do not apply in obvious ways. For example, mouse movement is often recorded in the form of XY coordinates on a Cartesian plane alongside timestamps, captured at a set polling rate, generally measured in hertz (e.g., 60 observations per second = 60 Hz). A single set of coordinates reflecting mouse position at a single point in time gives us nearly zero information about any potential construct of interest. In contrast, 60 Hz spatiotemporal data captured over 10 minutes, producing a total of 36,000 observations per person, might provide value in aggregate if engineered appropriately, as will be described in Step 5.

Given this change, firm theoretical grounding for meaningful groups of variables, rather than individual item content, serves as primary content-related validity evidence for trace models. In the case of digital spatiotemporal interaction data, information theory has provided some foundation for the relevance of mouse movement to a variety of constructs, although significant technical and domain knowledge is required in both information theory and theories of cognition to develop a well-described model. As noted by Fischer and Hartmann (2014), psychological traits often manifest themselves spatially, yet psychologists have rarely processed and analyzed spatial data due to its relatively high complexity in relation to traditional psychological measurement. In this context, techniques from information theory, signal processing, and classical statistics must be combined to enable interpretation and use of such data, work that is in early stages. What is understood now is that the behaviors that create trace data are the result of a complex longitudinal human system of decision making and unconscious reactions, themselves the outcomes of a complex network of situational stimulus (i.e., assessment content) by individual difference interactions. By conceptualizing spatiotemporal interaction data as longitudinal, new variables reflecting distances, velocities, and accelerations can be created, and these engineered variables have shown predictive value (Zgonnikov et al., 2017). Put plainly, a person's mouse location at a particular 1/60th of a second means little to nothing, but the speed and angle they move between relevant digital objects within an assessment may reflect individual difference constructs when aggregated and transformed meaningfully. Further research is needed to strengthen this foundation and provide a more precise roadmap for such engineering.

Step 2: Trace data architecture

Meaningfully building models from trace data requires that the original data have certain characteristics that enable the feature engineering pipeline suggested by prior theory and practice. In the case of spatiotemporal interaction data, meaningful calculation of distances, velocities,

accelerations, and other such features from XY coordinate data require participant identifiers and timestamps for each coordinate pair. Because observations can occur faster than timestamps can differentiate (e.g., if the timestamp is recorded by second but the polling rate is 30 Hz, there will be 30 measurements within a single second that incorrectly appear to have occurred simultaneously), a number capturing the sequence of events is also central. Further, specific trace data analytic systems have specific data input requirements. For example, in R, the *mousetrap* package requires all of these variables to engineer new, more psychologically meaningful variables (Kieslich, 2019). Not all trace data capture systems record the necessary information for all engineering processes, so the quality of initial trace data should never be assumed. Trace data are never automatically useful for measurement.

Another architectural concern regards any mismatch between the original purpose of trace data collection and the goals of an assessment expert's secondary analysis. Often, trace datasets are created as part of a debugging process, so that if a user of the system reports a bug, programmers have adequate data to retrace the behaviors that led to that bug so that they can reprogram the system to remove it. Thus, the data format of trace data is generally optimized to programmer needs. This can create unexpected problems related to the multilevel nature of trace data observations. For example, consider a web-based multiple-choice assessment that occurs across multiple "pages" of data collection; there is no guarantee that an existing trace data capture system differentiates between pages. Although there may be good conceptual reasons from a psychological point of view that mouse movement speed during a cognitive ability measure and mouse movement speed during a personality measure may reflect different constructs, such reasoning is unlikely to have been present during the design of trace data capture system unless a psychologist was part of those discussions. This can make post-hoc identification of useful trace datasets challenging, so it is recommended that psychologists provide input when trace data capture systems are being designed if there is any possibility their data will be used as model inputs later.

Step 3: Development study design

Trace data construct models rely upon supervised machine learning, a term used to describe models created to predict a known criterion from a set of predictor variables using an existing dataset. Ordinary least squares (OLS) regression can be considered one of the simplest examples of a supervised machine learning model, although not all supervised machine learning models involve regression as statisticians define the

term. To create a model, a dataset must be identified or collected containing both a measure of the construct being targeted for measurement and the trace data that will be used to model it. Requirements for the design of this study are like those of a traditional assessment development study (see American Educational Research Association et al., 2014). Most centrally, the sample should be ideally collected at random from the population in which the model will be used, raising familiar concerns about the choice between concurrent versus predictive validation approaches, range restriction, and convenience sampling. Auer et al. (2022), for example, modeled data collected from a concurrent validation study and a convenience sample to build trace data models of personality and cognitive ability. This sample was likely range restricted due to college student selection effects, and the concurrent nature of the validation approach likely created internal validity threats, potentially attenuating prediction from their trace model, just as they would for any other measure validated similarly.

Step 4: Training/test split

In machine learning, a common technique to evaluate overfitting is to incorporate multiple mathematical validation approaches as standard steps in model-building to better estimate the ability of those models to generalize to unseen data. One of the most common of these is *holdout validation*, which refers to the practice of randomly splitting the data to create two datasets with independent cases, commonly referred to as the training and test datasets. In the language of data science, the term *training dataset* refers to any dataset from which models are created; in a traditional assessment validation project in which a job performance criterion is predicted from items or measure scores, the entire dataset would be considered the training set. Unlike traditional assessment validation, however, machine learning often involves several modeling techniques prone to overfitting. Thus, it has become common practice to split the original dataset into two pieces, one to be used for “training” and the other to be used for “testing.” When training/test splits are done for the purpose used here, this technique is referred to as *holdout validation*. In this approach, the training set is used for modeling, and predicted scores using that model in the test set are used to evaluate final model performance.

We highlight two technical concerns that assessment experts may find surprising related to the specification of a training/test split. First, the best balance between the sample size in the training and test sets is not a settled theoretical issue, although there are common practices. Most commonly 80-20 splits are used; however, this decision may be affected

by sample size considerations. For example, if an 80-20 split would result in a test set too small to create a stable estimate of model performance, an alternative split may be used (e.g., 70-30) or an alternative training/test strategy entirely may be adopted. Second, it is important to conduct this split before any modeling or data exploration is attempted to avoid *leakage*, which refers to casewise non-independence between the training and test sets and therefore a contaminated estimate of model performance when estimated using the test set. Importantly, hierarchical structures within the data must be split such that higher-order groups of lower-order observations remain together once the training and test datasets have been generated. Most commonly, this requires that the split be done at the person level and that all within-person observations are allocated to the dataset where their associated person was randomly assigned. Otherwise, data within persons will be split across training and test, which would be considered a type of leakage.

Step 4 is the last step before the core iterative engineering loop. Steps 5–8 are likely to be repeated multiple times as the model is built and refined. Thus, to reduce the risk of overfitting in the final model, as of the split, the test set should not be used or referenced whatsoever until Step 9.

Step 5: Data wrangling

Trace data are commonly recorded to *event logs*, which are datasets containing *events*, a data format likely unfamiliar to assessment experts. Events can include spatiotemporal data if such data were of interest to the programmer, but it is safer to assume that event logs simply contain whatever data a programmer wanted to record longitudinally for whatever reason they wanted to record it. Event logs are typically in a format similar to the one seen in Table 2.

Individual events contain very little useful information in their raw form. The first line of Table 2, for example, indicates that user 123's

Table 2. Pseudocode example of trace data in an event log, based upon Auer et al. (2022) data.

PID	Seq	Timestamp	Event	Content
123	1	2020-10-04 14:10:04	MP	{ y:0; x:0 }
123	2	2020-10-04 14:10:04	GL	{ sessionId: b3d8d249; game: 1 }
123	3	2020-10-04 14:10:04	MP	{ y:10; x:20 }
123	4	2020-10-04 14:10:05	MP	{ y:10; x:20 }

Note. PID = Participant identifier; Seq = Sequence number; MP = Mouse position, GL = Game load event. The original data structure and format observed in the Auer et al. (2022) dataset has been simplified to illustrate multiple events occurring within the same second and general data format. Nesting within Content is also simplified but is shown in the original textual format, which is called JavaScript Object Notation.

mouse cursor was located at origin, which in web applications is usually the top-left corner of the web browser's main display area. For such data to be useful, they must be transformed into new variables related to the intended construct to be measured, preferably based upon the theory identified in Step 1.

This conversion process is called *data wrangling*. Much as in previous steps, there is no single established procedure for data wrangling, although three iterative sub-steps are common. First, *pre-processing* refers to the alteration or transformation of raw data streams, often text, or otherwise high-complexity unstructured data into variables or variable sets more likely to be useful for modeling or more representative of distinct, content-valid variables. In the raw spatiotemporal data in Table 2, for example, the text string in the "Content" column would need to be converted into meaningful variables, such as a distinct "x" and "y" variables for rows containing "MP" events. Second, exploratory data analysis is used to confirm and expand both theoretical and practical understanding of the dataset. For example, once distinct X and Y variables have been created from relevant Table 2 events, an analyst might decide to examine the distribution of mouse clicks per page or over several distinct time spans to assess if there is sufficient variance or complexity in the resulting summaries for inclusion in later models.

Additionally, it is important at this stage to assess missingness, especially to determine if missingness is construct-relevant and worth modeling. Third, *feature engineering* refers to the conversion of pre-processed data into variables for inclusion in the final model (Zheng & Casari, 2018). The most effective feature engineering processes for trace data will be done heavily informed or derived directly from the theory identified in Step 1; specifically, high quality features are most likely to result from a content domain expert constructing and testing working hypotheses based upon their expertise. In the questionnaire trace example, features might be created reflecting speed of mouse movement toward or away from meaningful points on the webpage, which in turn might be intended to reflect hesitation, confusion, or other state variables related to traits of interest.

Data wrangling is generally the most time-consuming step of trace modeling, because it is an iterative process requiring the conversion of complex event data into meaningful and properly structured features as inputs for machine learning. High quality feature engineering requires creativity, rigorous testing, and extensive documentation. Features are frequently developed, investigated deeply, but then abandoned. Only a relatively small subset of tested features will generally be included in the next step. This is normal part of the engineering process (Landers & Marin, 2021).

Step 6: Algorithm and hyperparameter identification

Even with cautious and thoughtful feature engineering, the predictors in the resulting models still generally correlate quite weakly with construct, necessitating the more complex predictive modeling approach of machine learning. Machine learning is a broad concept but can be conceptualized as a customization or evolution of statistical methods to address a class of prediction problems in which traditional statistical methods generally underperform. At significant risk of oversimplifying, most statistical methods familiar to people with psychometric assessment expertise generally prioritize unbiased estimation of model parameter values, such as regression weights, whereas machine learning methods generally prioritize maximizing generalizable predictive accuracy, such as out-of-sample R^2 (Yarkoni & Westfall, 2017). The balance between these goals, often called the *bias-variance tradeoff*, characterizes a key tension actively managed in machine learning; adding bias can increase out-of-sample generalizability when models are complex or when many or most assumptions associated with the standard statistical tests are violated (Hastie et al., 2009). Thus, machine learning methods are generally preferred in situations like typical trace data analysis, where a relatively small sample (e.g., $N = 500$) might be used to model a very large number of predictors (e.g., $k = 25000$). Whereas an OLS regression analysis would likely reveal an extremely high R^2 in such a situation due to the wealth of information available to triangulate on every individual case, that R^2 would be upwardly biased and unlikely to generalize to other samples.¹ Model complexity and bias can be so high that it becomes difficult or practically impossible for a human to interpret parameters.

¹An accessible entry-point to understand how the addition of bias to OLS regression can increase out-of-sample predictive accuracy is found in ridge regression, which is at its core OLS regression but with one change. Whereas OLS regression is used to derive the line of best fit in relation to the minimum residual sum of squares, ridge regression identifies the line of best fit in relation to the sum of the minimum residual sum of squares and the sum of the model's squared regression parameters. By doing this, larger regression coefficients are "penalized", causing models with smaller parameter values to appear superior to models with larger parameter values, which are common when model complexity, and especially multicollinearity, is high. Although the resulting *regularized* regression parameters are now biased by the weight of the penalty term, this change tends to reduce variance in out-of-sample predictive accuracy across models (Gibbons, 1981). In this way, the practice of adding bias to OLS regression in the form of a model complexity penalty term decreases the variance of out-of-sample performance (Tibshirani, 1996). Importantly, if a trace data model could be sufficiently simplified such that a small number of engineered features could be used to meaningfully predict an outcome of interest, it is unlikely that machine learning would outperform OLS regression in that context given the lower model complexity. However, we are unaware of any cases where such a low complexity trace model has been used to meaningfully predict an assessment-relevant outcome of interest.

This is sometimes called *the black-box problem* (Castelvecchi, 2016), a major concern in trace data modeling that leads to the content validity strategy outlined in Step 1 rather than one relying upon interpretation by subject matter experts.

Machine learning algorithms should be chosen given the nature of the data to be modeled, with issues to consider including scale of measurement, data sparsity, sample size, and analyst skill, among others. Commonly, the first decision is between *regression models*, which in the machine learning literature refers to any supervised machine learning approach modeling a continuous outcome, and *classification models*, which refers to any supervised machine learning approach modeling a discrete outcome. Although many popular algorithms have been adapted to be applied in both cases (e.g., support vector machines for classification versus support vector regression), a major implication of this decision is the family of model performance statistics that are used later. Whereas regression models are typically evaluated using R^2 and root mean square error (RMSE), classification models rely on a family of more unfamiliar metrics to psychometric assessment experts, such as area under the receiver operating curve and F1 scores. Because existing construct measures that might be used as criteria in trace models, such as cognitive ability or personality test scores, tend to be continuous, the remainder of this discussion will focus on regression models.

Once a decision is made between regression or classification models, a set of specific algorithms for testing are chosen. Algorithm choice should generally be based upon the expected performance of the algorithms given the dataset. For example, random forests and newer algorithms derived from it, like extreme gradient boosted trees (Hastie et al., 2009) are in part optimized to uncover high-complexity interactions, so such a model might be chosen if such interactions were expected. Further, it is common to include relative simpler algorithms, like elastic net regression (Zou & Hastie, 2005), for comparison. Because elastic net regression is essentially identical to OLS regression but with the addition of penalty terms to penalize inflated model parameters, it is easier to interpret elastic net model parameters than the model parameters from something as complex as extreme gradient boosted trees. This makes comparisons of error metrics like RMSE between elastic net and other more complex models a common tool for identifying best-performing algorithms, to weigh any increased performance gains in prediction from consideration of non-linear or otherwise difficult to interpret relationships against reductions in explainability. Generally, techniques with greater ability to detect and model subtle and high-complexity predictor-criterion relationships also require greater sample sizes, another key consideration; techniques commonly falling under the label of “deep learning,” for

example, often require tens of thousands of cases and benefit from millions or more. Putka et al. (2018) provide greater detail on the relatively simpler algorithms common in assessment contexts, but their list should be considered neither ideal nor exhaustive; as with all steps, selecting an algorithm should be part of the iterative engineering process of Steps 5-8. However, as a starting point, we typically recommend elastic net regression as a baseline, extreme gradient boosted trees or another high-complexity approach with larger sample size requirements, and one or two others suitable to the data set at hand.

Once algorithms have been identified, relevant hyperparameters and their values for testing must be identified. Conceptually, hyperparameters are “settings” for algorithms that are not adjusted during model training, and different algorithms have different settings. For example, elastic net regression penalizes inflated model parameters using two techniques called L1 regularization, which penalizes the sum of the absolute values of regression parameters, and L2 regularization, which penalizes the sum of the squared regression parameters. To control these penalties, elastic net regression has two hyperparameters. The first, commonly called α , defines the split of the overall penalty between L1 and L2. At $\alpha=.5$, the penalty will be 50% L1 and 50% L2. The second hyperparameter, commonly called λ , defines the strength of the penalty.

Importantly, there are few situations where the “best values” for these hyperparameters are known before modeling begins. Instead, Step 7 will involve testing a range of potential hyperparameter values to identify which leads to the best model performance—in the regression case, the largest R^2 .² Thus, at Step 6, the goal is to identify a reasonable range of hyperparameter values to test. In the case of elastic net, because α is a proportion, a uniform set of proportions between 0 and 1 might be chosen. λ has no standardized scale but is commonly split into 100 or more pieces between 0 and 5. This approach to testing of a range of plausible values across multiple hyperparameters simultaneously is called a *grid search*; for example, if 10 values of α and 100 values of λ are examined, a grid of 1000 combinations of those parameters will ultimately be tested. In contrast, extreme gradient boosted trees have five altogether different hyperparameters and reasonable ranges for them; thus hyperparameter lists and plausible ranges are unique to every algorithm and sometimes to the algorithm and dataset in combination.

Step 7: Model fitting and k-fold cross-validation

In this step, model performance for all combinations of selected algorithms and their hyperparameters are tested. To prevent overfitting given the high degree of data-driven decision making in this step, a technique

Table 3. Examples of hyperparameter crossing and resulting model counts during k-fold cross-validation.

Algorithm	Folds	Hyperparameters	Models Per Fold	Models Fitted
Elastic Net	10	2: alpha, lambda	$10 * 100 = 1000$	10000
XGBoost	10	5: eta, max_depth, colsample_ bytree, subsample, nrounds	$2 * 10 * 2 * 10 * 10 = 4000$	40000

called *k-fold cross-validation* is used at this stage. In k-fold cross-validation, the training set is split at random into *k* independent folds, commonly 5 or 10. For example, a 1000-case dataset subjected to 10-fold cross-validation would be split into ten 100-case folds. Next, *k* analyses are conducted for each unique algorithm and hyperparameter pair by fitting a model using *k*-1 folds as a temporary training set and the held-out fold as the test set. In the present example, an N=900 dataset would be used for model fitting, and model performance would be measured by applying that model to the held-out N=100. This process is then repeated across all fold combinations, and mean model performance is calculated across folds. Thus, for a single set of hyperparameter settings in 10-fold cross-validation, 10 models will be created with 10 different temporary training sets and 10 temporary test sets. This process is then repeated for each combination of hyperparameter settings in the grid, and in the case of regression, the mean and standard deviation of R² and RMSE across folds within settings are tested. An example with sample hyperparameter values and total model counts appears in [Table 3](#). The hyperparameter settings that produce the best model performance then becomes the “final” hyperparameter settings, and these settings are used to fit the final model using the original training set (not split into folds).

In regression, evaluating model performance is a matter of balancing two issues: mean/median performance and the distribution surrounding performance. For example, if one algorithm during k-fold cross-validation yields a higher mean R² than another but with a larger standard deviation of R² across folds, that inconsistency indicates increased risk for the generalizability of that model across future contexts. It is ultimately up to the analyst to balance risk and performance, if needed, depending upon the final purpose of the model, such as for research purposes only or for consequential decision-making about those being assessed.

Step 8: Model refinement and revision

As described earlier, model refinement is an iterative process. This may involve the creation of informative exploratory analyses and visualizations, changes to the features or modeling decisions, re-running models, and

observing the effects. Each iteration increases the risk of overfitting to the cross-validation set, so analyst skill plays an important role in mitigating the risk of poor cross-validation to the test set in the next step. The analyst should always consider the riskiness of any change or modeling decision and actively mitigate risk. For example, dropping an entire block of variables because none performed across folds is a relatively untargeted change and unlikely to harm generalizability substantially. However, identifying a subset of 5% of cases that based upon visualization seem to exhibit a “unique pattern” and engineering new features to reflect that pattern would be much riskier, although not necessarily unjustified. All such changes must be approached critically. Regardless, the analyst in this point should iterate through earlier steps until satisfied with the final model performance level achieved. Although possible, revisiting earlier steps *after* observing the results of Step 9 can increase the risk of overfitting in way not addressed with the use of k-fold cross-validation and other approaches outlined here. Thus, revisiting modeling decisions after Step 9 has begun should generally be considered a questionable research practice.

Step 9: Holdout validation and psychometric evaluation

Once a final k-fold cross-validated model has been selected, this model should be used to make predictions in the holdout set and test the accuracy of those predictions against the observed criterion values. As in previous steps, for regression-based models, this is typically done with R^2 and RMSE. By comparing the final k-fold cross-validated performance with holdout performance, the analyst can also assess the degree to which the cross-validated model is likely still overfitted. When final k-fold model performance in the training set is better, practically speaking, than performance in the holdout set, it suggests overfitting in the training set. When final k-fold model performance is similar or worse than holdout performance, it suggests that a generalizable model has likely been created. However, this can and should be later verified with additional data collection in true out-of-sample validation studies.

To evaluate if the model is of sufficient predictive accuracy to replace the construct measure that was modeled, the next testing goal is to estimate the correlation between the new predicted scores and a relevant criterion measure, which can be accomplished in two ways. [Figure 1](#) illustrates the general network of relationships to consider when the goal is to maximize the prediction of an organizational criterion with a given true score validity (a) and the existing construct measure's observed criterion relationship (b). The first and preferred strategy is direct validation, accomplished by measuring the relationship between the organizational criterion and predicted scores (c) and comparing this with the performance of the original construct

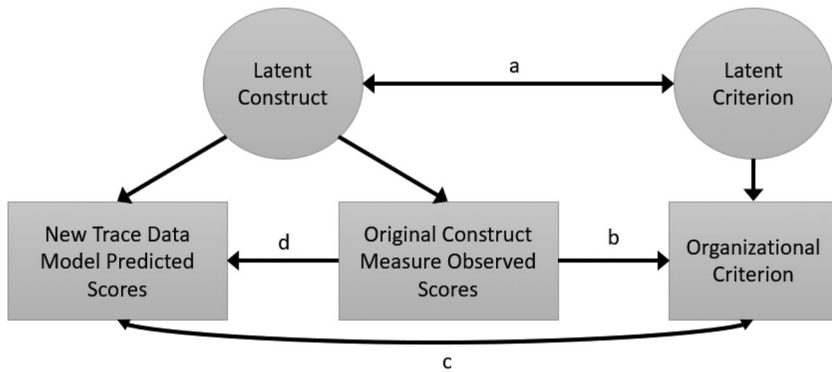


Figure 1. Relationships between construct, observed scores, and predicted scores in supervised ML-based trace data modeling.

measure (b). Generally, it is expected that relationship (c) will be smaller than relationship (b) due to attenuation caused by imperfect measurement reflected in the trace data model's performance (d). However, the trace data model might perform better than the original measure if it contains construct-irrelevant but criterion-relevant variance, a type of construct contamination. The second strategy can be used to estimate (c) without additional data collection if relationship (b) is already known, such as from a previous validation study, although this does assume no construct contamination in trace model predictions. This can be estimated using the standard correction for attenuation formula; Table 4 contains a convenient quick look-up table for this same information. For example, if an original personality measure predicts job performance at $r=.20$, and the trace model is able to achieve $r=.70$, the expected relationship between the trace model predicted score and job performance would be .17. With either strategy, it is now the designer's decision as to whether the level achieved was sufficient for practical application in a decision-making system.

Practical implications and applications

Despite the infrastructure and human resource requirements needed for trace analysis often beyond the traditional skill set of traditional testing professionals, the benefits may outweigh these additional practical requirements. We present three potential advantages and one caveat. First, the use of trace data may simply enable the collection of trait score estimates that would otherwise be unattainable. For example, in a lab setting during development, trace data could be modeled on complex assessments requiring individual live test proctoring or other complex and time-consuming methods. A trace model able to predict

Table 4. Predicted trace data model validity given model performance and observed criterion prediction.

Trace Model		Observed Criterion Prediction <i>R</i> by Original Construct Measure								
<i>R</i>	<i>R</i> ²	.05	.10	.15	.20	.25	.30	.35	.40	.45
.99	.98	.05	.10	.15	.20	.25	.30	.35	.40	.45
.95	.90	.05	.10	.15	.19	.24	.29	.34	.39	.44
.90	.81	.05	.09	.14	.19	.24	.28	.33	.38	.43
.85	.72	.05	.09	.14	.18	.23	.28	.32	.37	.41
.80	.64	.04	.09	.13	.18	.22	.27	.31	.36	.40
.75	.56	.04	.09	.13	.17	.22	.26	.30	.35	.39
.70	.49	.04	.08	.13	.17	.21	.25	.29	.33	.38
.65	.42	.04	.08	.12	.16	.20	.24	.28	.32	.36
.60	.36	.04	.08	.12	.15	.19	.23	.27	.31	.35
.55	.30	.04	.07	.11	.15	.19	.22	.26	.30	.33
.50	.25	.04	.07	.11	.14	.18	.21	.25	.28	.32
.45	.20	.03	.07	.10	.13	.17	.20	.23	.27	.30
.40	.16	.03	.06	.09	.13	.16	.19	.22	.25	.28
.35	.12	.03	.06	.09	.12	.15	.18	.21	.24	.27
.30	.09	.03	.05	.08	.11	.14	.16	.19	.22	.25
.25	.06	.03	.05	.08	.10	.13	.15	.18	.20	.23
.20	.04	.02	.04	.07	.09	.11	.13	.16	.18	.20
.15	.02	.02	.04	.06	.08	.10	.12	.14	.15	.17
.10	.01	.02	.03	.05	.06	.08	.09	.11	.13	.14
.05	.00	.01	.02	.03	.04	.06	.07	.08	.09	.10

Note. Trace model performance is (d) and observed prediction is (b) in Figure 1.

such scores using existing assessments could be later used without the expense of data collection with the original measure. Second, insights gained from large trace datasets could prove to complement more theory-driven scoring approaches, offering opportunities for enhanced measurement and predictive potential.

Some machine learning algorithms are more explainable than others (Gunning et al., 2019), and the choice to focus on such models during Step 6 enables the analysis of such explanations to learn more about how a construct is being measured or could be measured in future assessment revisions. Third, as the bulk of trace data comprise behaviors not obviously tied to measurement, the use of such models may make response distortion more challenging as trace-producing behaviors are generally difficult to predict and control. Beyond these advantages, a practical concern is the degree to which scores derived from trace data modeling are transparent enough to interpret and explain to organizational decision-makers. As controversial as both trace data and machine learning may be within the assessment community, understanding is even poorer among many traditional decision-makers outside of the technology sector. Putka et al. (2018) offer a number of strategies in this situation, such as the use of importance indices and visual representations, to assist in describing to lay decision makers how predicted values from such complex models have been derived. Nevertheless, this is an important barrier to adoption that should be addressed directly.

In terms of traditional psychometric concerns, we emphasize that (a) the generalizability of test scores to similar measures and (b) the accuracy of inferences made on test scores (Cronbach et al., 1972) are still paramount. As much as minimizing residual variance is a main objective of predictive modeling, minimizing error variance is a main objective of psychometrics. In other words, reliability is a fundamental psychometric property of test scores that psychometricians seek to maximize. Maximizing the reliability of scores requires researchers to determine multiple sources of true score variance and error variance (Shavelson et al., 1989). Because micro-behaviors are strongly influenced by many different factors, trace data generally only provide weak signals in true score variance of target constructs from a vast set of behaviors (e.g., mouse trajectories, clicking behaviors). It is only through meaningful aggregation of these weak signals into meaningful features that stronger signals of systematic true score variance can be developed. Because the data are inherently organic (see Xu et al., 2020, p. 1257), identifying sources of measurement error can be complex. However, as shown in Step 1, these weak signals can be motivated theoretically by considering their partitioning of reliable and unreliable sources of variance. For example, in attempting to model cognitive ability, Auer et al. (2022) identified and focused upon trace data likely reflecting cognition (e.g., fast and stable mouse movement directly to the correct answer on a validated cognitive assessment) while acknowledging noise related to cognitive load limitations on working memory due to screen size (see Arthur et al., 2018), lag time due to spotty internet connection, erratic mouse hardware, low clicking precision due to inexperience with computers and mice, browser incompatibility, and so on. Determining, isolating, and mitigating each of these sources of error variance and likely others is therefore recommended to improve prediction from trace data, increasing the validity of inferences, such as through standardization of the testing environment. Finally, all such efforts and their effects should be well-documented to provide a comprehensive picture of the assessment development process for later auditing (Landers & Behrend, [in press](#)).

Ethics of measurement with trace data

Most research and practical interest in trace data to date has been marketing and consumer behavior research rather than psychometrics. For example, when people use search engines to explore topics of personal interest, trace data harvesters can collect comprehensive histories of search term use, the dates and times of those searches, and the likely identities of the people doing the searching. These data are then often used as predictors in predictive models of purchasing behavior, which

can in turn be used to encourage those search engine users to spend money on specific targeted products and services, often in ways seemingly unconnected to the search engine. Because many internet users are aware of neither the extent to which their trace data are being collected and analyzed nor the extent to which these models are being used to manipulate their behavior, privacy researchers often highlight the questionable ethics of such data collection.

Trace data analysis on existing digital assessments appears less ethically problematic. First, assessees completing traditional digital assessments are generally aware that they are completing assessments. Whereas Google repurposes search data to model purchasing behavior, trace data modeling on existing digital assessments repurposes data collected for assessment for use in additional assessment. As such, there is far less potential for deception, because assessees already understand they are being assessed; adding trace data modeling only changes the technical details of how. If use of the original digital assessment could be considered ethical, adding trace data analysis changes little. Second, trace data analysis theoretically makes full use of the data provided by those being assessed. If such data can be used to make valid inferences about individual capabilities beyond those already assessed, other assessments could potentially be dropped, reducing assessee time, effort, and stress to get the same quality of information. In this way, failing to investigate the potential of trace data is potentially the more unethical approach.

Conclusion

In summary, we have provided here a process by which trace data can be identified, engineered, analyzed, and considered for use as psychometric measures. Although we focused here upon spatiotemporal data, this process can be applied effectively to other types of trace data. To be absolutely clear, we contend that most trace data as currently collected and given current, realistic analytic options are unlikely to reach a psychometric quality threshold that will justify dropping existing measures and replacing them with trace-based measures. Instead, we urge assessment professionals to consider two paths. First, trace data as it currently is collected is a potentially useful source of supplemental information that can be used to improve existing measures and in a much smaller number of ideal cases, replace them. By engineering models carefully and comprehensively, predicted scores from such models may be used to improve measurement quality in existing assessments or get additional weak reflections of constructs that could not otherwise be collected in the same amount of time. By then aggregating across trace and purposive models, assessment developers may be able to improve measurement

quality in previously unexplored ways. Attempting this with interdisciplinary teams consisting of data engineers, statisticians, psychologists, and assessment experts is likely to lead to the best results. Second, systems that are designed and developed to produce useful trace data are an entirely unexplored frontier, one which we contend has great potential to legitimately and validly replace existing measures in the long run. For example, if an assessment was designed explicitly with the purpose of making mouse trajectories and timings construct-relevant, trace data modeling of those trajectories and timings might be sufficient to reach meaningful measurement standards. It is only with continued experimentation with assessment software design and trace models, as well as through thoughtful interdisciplinary collaborations, that we will ever understand this potential.

ORCID

Richard N. Landers  <http://orcid.org/0000-0001-5611-2923>

References

- American Educational Research Association, American Psychological Association, & National Council on Measurement in Education. (2014). *Standards for educational and psychological testing*. American Educational Research Association.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., Zimmermann, T. (2019). Software engineering for machine learning: A case study. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)* (pp. 291–300). <https://doi.org/10.1109/ICSE-SEIP.2019.00042>
- Arthur, W., Jr., Keiser, N. L., Hagen, E., & Traylor, Z. (2018). Unproctored internet-based device- type effects on test scores: The role of working memory. *Intelligence*, 67, 67–75. <https://doi.org/10.1016/j.intell.2018.02.001>
- Auer, E. M., Mersy, G., Marin, S., Blaik, J., & Landers, R. N. (2022). Using machine learning to model trace behavioral data from a game-based assessment. *International Journal of Selection and Assessment*, 30(1), 82–102. <https://doi.org/10.1111/ijsa.12363>
- Castelvecchi, D. (2016, October 5). Can we open the black box of AI? *Nature*, 538(7623), 20–23. <https://www.nature.com/news/can-we-open-the-black-box-of-ai-1.20731> <https://doi.org/10.1038/538020a>
- Cronbach, L. J., Gleser, G. C., Nanda, H., & Rajaratnam, N. (1972). *The dependability of behavioral measurements: Theory of generalizability for scores and profiles*. Wiley.
- Fischer, M., & Hartmann, M. (2014). Pushing forward in embodied cognition: May we mouse the mathematical mind? *Frontiers in Psychology*, 5, 1315–1078. <https://doi.org/10.3389/fpsyg.2014.01315>
- Gibbons, D. G. (1981). A simulation study of some ridge estimators. *Journal of the American Statistical Association*, 76(373), 131–139. <https://doi.org/10.1080/01621459.1981.10477619>

- Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., & Yang, G.-Z. (2019). XAI— Explainable artificial intelligence. *Science Robotics*, 4(37), eaay7120. <https://doi.org/10.1126/scirobotics.aay7120>
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer-Verlag.
- Kieslich, P. J., Henninger, F., Wulff, D. U., Haslbeck, J. M. B., & Schulte-Mecklenbeck, M. (2019). Mouse-tracking: A practical guide to implementation and analysis. In M. Schulte-Mecklenbeck, A. Kühberger & J. G. Johnson (Eds.), *A Handbook of Process Tracing Methods*, (pp. 111–130). Routledge.
- Landers, R. N., Armstrong, M. B., Collmus, A. B., Mujcic, S., & Blaik, J. (2022). Theory- driven game-based assessment of general cognitive ability: Design theory, measurement, prediction of performance, and test fairness. *Journal of Applied Psychology*, 107(10), 1655–1677. <https://doi.org/10.1037/apl0000954>
- Landers, R. N., & Behrend, T. S. (in press). Auditing the AI auditors: A framework for evaluating fairness and bias in high stakes AI predictive models. *American Psychologist*, Advance online publication. <https://doi.org/10.1037/amp0000972>
- Landers, R. N., & Marin, S. (2021). Theory and technology in organizational psychology: A review of technology integration paradigms and their effects on the validity of theory. *Annual Review of Organizational Psychology and Organizational Behavior*, 8(1), 235–258. <https://doi.org/10.1146/annurev-org-psych-012420-060843>
- Murphy, K. R., & Aguinis, H. (2019). HARKing: How badly can cherry-picking and question trolling produce bias in published results? *Journal of Business and Psychology*, 34(1), 1–17. <https://doi.org/10.1007/s10869-017-9524-7>
- Putka, D. J., Beatty, A. S., & Reeder, M. C. (2018). Modern prediction methods: New perspectives on a common problem. *Organizational Research Methods*, 21(3), 689–732. <https://doi.org/10.1177/1094428117697041>
- Shavelson, R. J., Webb, N. M., & Rowley, G. L. (1989). Generalizability theory. *American Psychologist*, 44(6), 922–932. <https://doi.org/10.1037/0003-066X.44.6.922>
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1), 267–288.
- Xu, H., Zhang, N., & Zhou, L. (2020). Validity concerns in research using organic data. *Journal of Management*, 46(7), 1257–1274. <https://doi.org/10.1177/0149206319862027>
- Yarkoni, T., & Westfall, J. (2017). Choosing prediction over explanation in psychology: Lessons from machine learning. *Perspectives on Psychological Science : A Journal of the Association for Psychological Science*, 12(6), 1100–1122. <https://doi.org/10.1177/1745691617693393>
- Zgonnikov, A., Aleni, A., Piironen, P., O'Hara, D., & Di Bernardo, M. (2017). Decision landscapes: Visualizing mouse-tracking data. *Royal Society Open Science*, 4(11), 170482. <https://doi.org/10.1098/rsos.170482>
- Zheng, A., & Casari, A. (2018). *Feature engineering for machine learning: Principles and techniques for data scientists*. O'Reilly Media, Inc.
- Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 301–320. <https://doi.org/10.1111/j.1467-9868.2005.00503.x>