CHAPTER 1

Machine Learning Algorithms and Measurement

Q. Chelsea Song, Ivan Hernandez, Hyun Joo Shin, Meaghan M. Tracy, and Mengqiao Liu

MACHINE LEARNING AND ITS UNIQUE CONTRIBUTION TO MEASUREMENT

Machine learning (ML) is a subfield of artificial intelligence that utilizes data to optimize predictions and discover underlying patterns (e.g., Mitchell, 1997). Compared to traditional approaches for measuring human attributes (e.g., classical testing theory, item response theory), machine learning uniquely contributes to measurement by being better suited to (1) utilize organic data and (2) capture complex relations. Organic data are naturally occurring digital footprints that are collected without reliance on a specific research design or measurement scale; examples include online search records, Twitter posts, and location data collected from fitness trackers (see Groves et al., 2011; Hickman, Bosch, et al., 2022; Xu et al., 2020). Such data convey rich behavioral and psychological traces embedded in everyday contexts, providing valuable information for measurement. However, due to their complexity and lack of structure, they were rarely utilized in psychological measurement – until the introduction of machine learning. With machine learning, we are now capable of analyzing a diverse and complex range of data, from self- and other-reports to audiovisual footprints. To name a few, machine learning is used to measure personality from interview videos (e.g., Hickman, Bosch, et al., 2022), stress and emotions from social media posts (e.g., Wang et al., 2016), and interpersonal relationships from proximity data obtained from wearable sensors (e.g., Matusik et al., 2019). Such capability allows for increased accuracy in measurement as well as ecological momentary assessment of human behavior and cognition – enabling a more comprehensive measurement of human psychology and behavior.

Machine learning can also capture complex relations (e.g., nonlinear relations and interactions), potentially uncovering new insights into

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

psychological phenomena. Recent works utilized machine learning to study personality nuances at the facet and item levels, extending the theoretical and practical understanding of personality traits (e.g., Putka et al., 2018). Unlike traditional parametric methods, ML algorithms do not rely on a priori specification of dimensions, allowing for a more flexible examination of the data (e.g., Jiang et al., 2020). Finally, ML algorithms are capable of handling high-dimensional data (where the number of features is large relative to the sample size), enabling the integration of multiple and complex data types while seeking to maintain the accuracy and generalizability of measurement.

In general, machine learning contributes to measurement in two crucial ways: conceptualization of a construct (via unsupervised learning) and empirical keying (via supervised learning). Unsupervised learning aims to find structure or patterns within data, and it could be used to explore the structure of a construct (e.g., to identify depressive symptoms among a wide variety of mental health symptoms). Similar to factor analysis in classical testing theory, unsupervised learning contributes to the conceptualization of a construct, providing the foundation for measurement. Yet, compared to traditional measurement methods, unsupervised learning has the advantage of identifying patterns from a large set of unstructured data, potentially contributing to broadened conceptual understandings. Supervised learning aims to estimate certain psychological constructs (e.g., ability) from a set of features (e.g., interview transcripts, event logs). Supervised learning, when used in measurement, is effectively an empirical keying method to convert features (or variables) into construct estimates – similar to the empirical keys used in traditional measurement, yet with improved accuracy, scalability, and consistency.[1] Together, machine learning contributes to measurement through conceptualization and empirical keying.[2] In the sections below, we discuss common ML algorithms used in measurement.

[1] A construct estimate could be continuous (e.g., ability level) as well as categorical (e.g., depressive symptoms). When supervised learning is used to measure a continuous construct, the process is called *regression*; when it is used to measure a categorical construct, the process is termed *classification*.

[2] The use of unsupervised (contextualization) and supervised learning (empirical keying) ML algorithms in measurement are guided by different approaches, which vary on the theory-data spectrum. Hickman, Song and Woo (2022) provides a systematic discussion of these approaches, which include (1) the theory-driven, hypothetico deductive approach, (2) the construct-driven, data-flexible approach, and (3) the data-driven, construct-informing approach.

1.1 Overview of Common ML Algorithms Used in Measurement

1.1.1 General Procedures of Using ML Algorithms in Measurement

A variety of ML algorithms can be used for measurement, yet their applications follow the same general procedure, which we describe in this section.

Suppose we want to measure the conscientiousness facets that are most relevant to one's performance in a certain job. We start by using unsupervised learning to define the construct and determine its structure (i.e., conceptualize the construct). For example, we ask incumbents to describe characteristics that help them successfully perform the job, and use unsupervised learning (e.g., topic modeling) to identify the common themes mentioned in the textual descriptions. Suppose the analysis identifies two main themes – dutifulness and order.

Next, we use supervised learning to develop empirical keys for measuring dutifulness and order. To do this, we begin by collecting data for model training and evaluation. Supervised learning requires two types of data: features and "ground truth." Features are variables used to estimate the construct. For instance, features for dutifulness and order include video recordings of an individual completing a work sample, email correspondence and log files related to certain work tasks, and human resource records on attendance – they all convey behavioral and psychological traces that reflect the construct. "Ground truth" is an operationalization of a construct provided by an existing, valid measure.[3] For instance, the "ground truth" of dutifulness and order could include self-reported facet-level scores from the NEO Personality Inventory-Revised (Costa & McCrae, 1992).

To train and evaluate an ML model, we split the dataset into two parts: training and test sets. The training set is used for training the model (or finding the empirical keys), and the test set is used to evaluate the performance of the model.

During model training, we first select an ML algorithm, and use the training data to find an optimal set of parameters that most accurately estimates the "ground truth" from the features. Hyperparameter tuning and cross-validation are used to find the optimal parameters, and the resulting model is called the trained model. For example, suppose we want to train an elastic net ML model to measure individuals' dutifulness and order. The elastic net model includes a number of parameters (e.g., feature

[3] The choice of "ground truth" is critical for construct validation. For a discussion of this issue, see Braun and Kuljanin (2015) and Tay et al. (2020).

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

weights) whose value could be adjusted to more accurately estimate the construct. We use hyperparameter tuning and cross-validation to systematically review a range of potential values for the parameters and identify the best set of values that provide the construct estimates most closely resembling the "ground truth" (e.g., self-reported dutifulness and order). These selected parameter values are used to specify the trained model.

Following model training, the next step is to evaluate the model using a test set. For example, we use the trained model to estimate individuals' dutifulness and order from their task completion log files and compare the estimates with self-reported dutifulness and order (the "ground truth"). If the model estimates approximate the "ground truth" well, the model passes the evaluation. The final model could be used to measure dutifulness and order in new samples.

In the following section, we describe the common ML algorithms used for measurement. The algorithms, as well as their example applications, are summarized in Table 1.1.

1.1.2 *Unsupervised Learning Algorithms*

1.1.2.1 *Clustering Methods*

Clustering methods are a type of unsupervised learning that seeks to discover distinct groupings (e.g., types of symptoms, groups of people). Clustering has been used to conceptualize collaborative problem-solving skills from log data of online simulation tasks (Polyak et al., 2017) and identify depressive symptoms from self-rated and clinician-rated depression scale scores (Chekroud et al., 2017).

k-Means Clustering: *k*-means clustering is one of the earliest and most commonly used clustering algorithms (MacQueen, 1967; Steinhaus, 1956). It first calculates the "mean" of *k* random sets of observations and uses them to determine *k* initial center points. Then, for each initial center point, the algorithm identifies a cluster of data that is closest to the center point and updates the location of the center point based on the mean of the new cluster. The algorithm iterates through these steps until the total within-cluster variance is minimized and the center points no longer change substantially (Hastie et al., 2009).

1.1.2.2 *Topic Modeling*

Topic modeling is used to find recurring semantic patterns among texts. One of the most popular topic modeling approaches is latent

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

Table 1.1 *Description of common ML algorithms and their example measurement applications*

Algorithms	Brief description	Example	Resources
Unsupervised learning			
k-means clustering	Calculates the "mean" of *k* random sets of data to determine *k* initial center points. For each initial center, the algorithm identifies a cluster of data that is closest to the center point, and repeatedly updates the location of the center point until the total within-cluster variance is minimized and the center points are stable.	Classify collaborative problem-solving skills of individuals from log files of online simulation (Polyak et al., 2017)	Steinhaus (1956)
Topic modeling	Identifies themes within documents. A common topic modeling algorithm, LDA, uses the distribution of the keywords in the documents.	Identify facets of job satisfaction from textual reviews (Jung & Suh, 2019)	Blei et al. (2003)
Supervised learning			
Linear regression			
OLS regression	OLS regression models the linear relation between the features and the outcomes of interest by minimizing the squared differences between the estimated and observed outcomes (i.e., the residuals) in the training data.	Measure dark side of personality from social media status updates (Akhtar et al., 2018)	Kenney and Keeping (1962)
Regularization			
Ridge	Ridge regression aims to reduce model overfit by shrinking feature weights *toward* zero without fully eliminating them through regularization.	Measure Big Five personality traits from Facebook profile status messages (Park et al., 2015)	Hoerl and Kennard (1970)

Table 1.1 (*cont.*)

Algorithms	Brief description	Example	Resources
LASSO	Similar to ridge regression, however, LASSO allows certain feature weights to be shrunk *to* zero, enabling both feature selection and regularization (shrinking *toward* zero).	Measure HEXACO personality from Facebook activities (Youyou et al., 2015)	Tibshirani (1996)
Elastic net	Improves upon ridge and LASSO regression by including both the penalty terms so that feature weights are shrunk both toward zero (regularization) *and* to zero (feature selection) and allows adjustment of the strength of regularization and feature selection.	Measure Big Five personality from automated video interviews (Hickman, Bosch, et al., 2022)	Zou and Hastie (2005)
Nearest neighbor			
k-NN	Classifies or estimates the value of an unknown observation through its nearest neighbors whose values are known. The final classification or estimate is determined by majority vote or through averaging.	Measure likelihood of a user spreading disinformation on social media by comparing their writing style to other known disinformation spreaders (Cardaioli et al., 2020)	Fix and Hodges (1951)
Tree-based models			
Decision trees	Partitions (or splits) data using if-then decision rules determined by the features. The feature that accounts for the most unexplained variance is introduced first, followed by the next most discriminatory feature, and so on.	Measure leadership style preference using educational degree, major, gender, and marital status (Salehzadeh, 2017)	Quinlan (1986)

Bagged trees and random forest	Bagged trees: trains multiple decision trees with bootstrapped samples from the training data. Random forest: trains multiple decision trees with bootstrapped samples, but the features of each decision tree are introduced at random. For both algorithms, the trained trees are separately used to estimate construct scores, and the final construct estimate is the average across (regression) or majority vote of (classification) the decision trees estimates.	Measure job performance from performance appraisal narratives from supervisors (Speer, 2020) Measure suicide intention from individuals' health care registry data (Gradus et al., 2020)	Breiman (1996) Breiman (2001)
Gradient-boosted trees	Trains decision trees that improve upon each other by learning from the previous decision tree. Stochastic gradient-boosted trees use random subsets from the training data to improve estimation.	Measure the degree that a job applicant taking a personality assessment was inflating their true personality trait scores (Calanna et al., 2020)	Friedman (2001, 2002)
Support vector machine	Estimates construct scores from linear and nonlinear relations by finding the optimal hyperplane (or threshold) that denotes the separation between data of different classes (classification) or the best approximation of the relation between outcomes and features within a given margin of error (regression).	Measure emotion from electroencephalography signals (Hassanien et al., 2018)	Drucker et al. (1996)
Neural network	Consists of complex interconnected structures that allow discovery of patterns and underlying relations in a set of data.	Measure psychopathy from tweets (Ahmad et al., 2020)	Hanson and Salamon (1990)

Dirichlet allocation (LDA; Blei et al., 2003), which aims to find the thematic structure within text documents. LDA identifies the theme of the documents based on the distribution of the keywords included in the documents (Chaney & Blei, 2012). Other topic modeling approaches include structural topic models (Roberts et al., 2014), which take into account covariates when creating topics; hierarchical LDA (Blei et al., 2003), which identifies topic hierarchies; and Top2vec (Angelov, 2020), which converts each of the N documents into a series of K numbers representing their standing on a variety of underlying dimensions (i.e., a document embedding, e.g., Doc2Vec, Universal Sentence Encoder, BERT) and applies clustering methods to the $N \times K$ data matrix.

Example Applications of Unsupervised Learning Algorithms

Clustering: Polyak et al. (2017) used *k*-means clustering to identify profiles of collaborative problem-solving skills exhibited in an online simulation game scenario. In the online game, players collaborate with an automated virtual agent to complete various missions (e.g., solve a passcode to unlock a door, discover a sequence of power transfer steps). The players engage in dialogues with the virtual agent, where in each dialogue, the virtual agent provides a prompt, and the player reacts by choosing a response among multiple options. The player's response choices, as well as other behavior data (e.g., number of mouse clicks, keystrokes, dialog selection timing) are recorded in a log file – this log file provides organic data with behavioral traces that reflect the player's collaborative problem-solving skills. The researchers conducted *k*-means clustering analysis on the log files and identified different profiles of collaborative problem-solving skills among players. These profiles allow researchers to further develop measures (e.g., games, response options) that better capture a comprehensive range of collaborative problem-solving skills.

In general, compared to traditional approaches where researchers manually code qualitative data to distill and categorize patterns (e.g., types of collaborative problem-solving skills), unsupervised learning methods offer more efficiency and scalability. This is especially important for measuring and studying collaborative problem-solving skills under new contexts, such as during remote and hybrid work. For example, researchers could analyze Slack (the online messaging application) data with the clustering method demonstrated in Polyak et al. (2017) to study how coworkers are collaborating with each other to solve problems during in-person and remote work settings.

Topic Modeling: Jung and Suh (2019) used LDA to identify facets of job satisfaction from 35,063 textual employee reviews posted on an online

job site (that capture employees' organic description of their jobs). The algorithm identified 30 different facets of job satisfaction, ranging from satisfaction toward organizational culture to work intensity and efficacy. Among these facets, some (e.g., satisfaction toward supervisor and pay) align with existing theory, while others (e.g., satisfaction toward project, interfirm relationship) provide new insights to existing literature. Topic modeling contributed to improved understanding and measurement of employee job satisfaction.

1.1.3 Supervised Learning Algorithms

1.1.3.1 Linear Regression

Linear regression models a linear relation between the features and the outcomes of interest. For example, in simple linear regression, feature X and outcome Y are modeled as: $y = B_0 + B_1X$, where B_0 represents the intercept of the regression line (i.e., the outcome value when the features are held constant at 0), and B_1 represents the slope of the regression line (i.e., feature weights, or how the outcome changes given one unit increase in a feature). Multiple linear regression consists of multiple features and one outcome and is expressed as: $y = B_0 + B_1X_1 + B_2X_2 + \ldots + B_nX_n$.

Ordinary Least Squares (OLS) Regression: OLS regression (Kenney & Keeping, 1962) aims to minimize the squared differences between the estimated and observed outcomes in the training data. This squared difference is termed the least squares error and is calculated as: $\sum_{i=1}^{N}\left(y_i - \widehat{y_i}\right)^2$, where $\widehat{y_i}$ is the estimated outcome obtained using the regression model and y_i is the observed outcome value. OLS regression aims to find a set of intercept and feature weights that minimize the cost function, and the resulting model provides the best linear representation of the relation between the feature and the outcome.

Although linear regression models are simple and commonly used for measurement, they are prone to model overfit (and high bias, as we discuss later). That is, linear regression models have the tendency to capture uniqueness in the training data, resulting in inaccuracy when they are used for measurement in new data or a new sample. For example, in Figure 1.1, Panel 1 shows a regression line (dashed line) that is trained with the training data (triangles). Although the regression line fits the training data well, as shown in Panel 2, it does not accurately model the test set

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

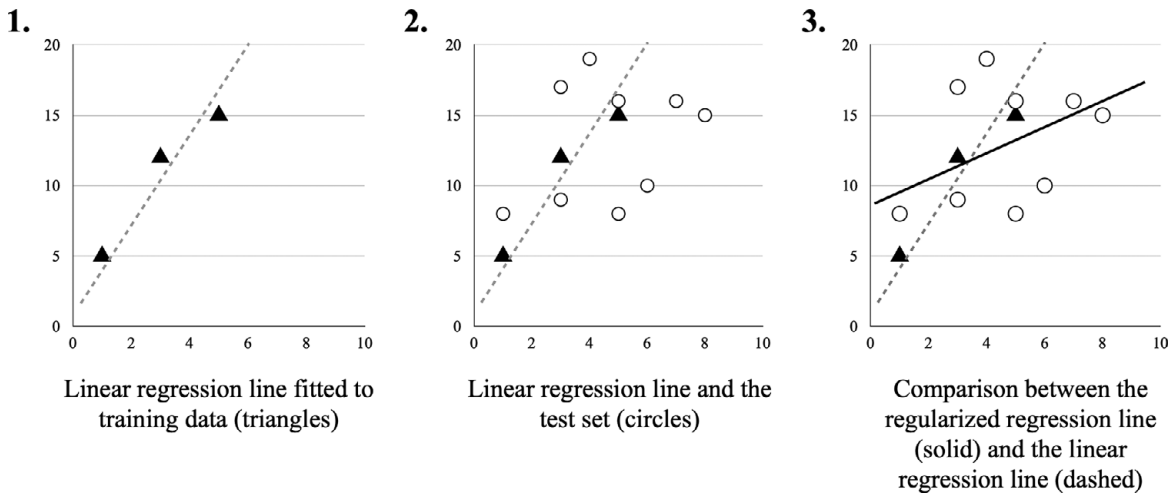

Figure 1.1 Comparison of linear and regularized regression models

(circles), suggesting that the regression model overfitted to the training data and will not perform well in a new sample.

1.1.3.2 Regularization

Regularization is a family of regression algorithms introduced to overcome the risk of model overfit. Figure 1.1, Panel 3 shows the regularized regression line (solid line). Compared to the overfitted linear regression line (dashed line), the regularized line fits well with both the training and test sets, resulting in higher estimation accuracy in the test set.

This improvement in regularization model accuracy is related to model bias and variance. Model bias refers to the systematic difference between the true population estimate and the model estimate; and variance refers to the variation in model accuracy when the model is applied to different sets of data. The combination of bias and variance determines the overall accuracy of a model: a model has higher accuracy when both the bias and variance are small. When training a model, we aim to lower both bias and variance. However, it is difficult to minimize both bias and variance in a model (a problem known as "bias-variance tradeoff"; Hastie et al., 2009). Regularization algorithms aim to increase estimation accuracy by striking an optimal balance between bias and variance. This is done through the regularization cost function, which includes the OLS least squares error term, a penalty term to regularize bias (and thus variance), and a tuning parameter to find an optimal balance between bias and variance. Regularization includes ridge (Hoerl & Kennard, 1970), LASSO (Tibshirani, 1996), LARS (Efron et al., 2004), and elastic net regression (Zou & Hastie, 2005).

Ridge Regression: Model overfit could be caused by multicollinearity (or redundancy among features) and large feature weights. Ridge regression aims to reduce model overfit by shrinking (regularizing) the feature weights (or regression coefficients) *toward* zero without fully eliminating them (Hoerl & Kennard, 1970). The ridge regression cost function is: $\sum_{i=1}^{N}\left(y_i-\widehat{y_i}\right)^2+\lambda\sum_{j=0}^{M}w_j^2$ where w is the feature weight for each j feature and λ is the tuning parameter that determines the regularization strength. The cost function includes a least squares error term, $\sum_{i=1}^{N}\left(y_i-\widehat{y_i}\right)^2$, and a ridge regression penalty term, $\sum_{j=0}^{M}w_j^2$. The penalty term adds to the overall error of the model, especially for large feature weights, and the strength of the regularization is adjusted by λ. As shown in Figure 1.2, the ridge regression penalty term (represented by the bolded

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

circle border) constrains the feature weights to a limited solution space (represented by the gray-shaded circle area). The intersection between the solution space (gray-shaded circle) and the least squares error (ellipse) provides a solution that minimizes the ridge regression cost function and strikes an optimal balance between bias and variance.

LASSO Regression: LASSO regression, similar to ridge regression, also aims to reduce model overfit; however, unlike ridge regression, LASSO allows certain feature weights to be shrunk *to* zero, enabling both feature selection and regularization (shrinking *toward* zero; Tibshirani, 1996). The LASSO cost function is: $\sum_{i=1}^{N}\left(y_i-\widehat{y_i}\right)^2+\lambda\sum_{j=0}^{M}\left|w_j\right|$, where the penalty term is $\sum_{j=0}^{M}\left|w_j\right|$. Compared to the ridge penalty term $\sum_{j=0}^{M}w_j^2$ (sum of squared coefficients), the LASSO penalty term is the sum of squared absolute coefficients, which allows the feature weights to be zero. As shown in Figure 1.2, the intersection between the solution space constrained by the LASSO penalty term (represented by the gray-shaded diamond) and the least squares error (ellipse) is where the feature weight, β_2, is zero, effectively dropping that feature.

Elastic Net Regression: Elastic net regression combines the characteristics of both ridge and LASSO regressions (Zou & Hastie, 2005). The elastic net cost function is: $\sum_{i=1}^{N}\left(y_i-\widehat{y_i}\right)^2+\lambda\left[\alpha\sum_{j=0}^{M}\left|w_j\right|+(1-\alpha)\sum_{j=0}^{M}w_j^2\right]$, and the penalty term consists of both the ridge and LASSO penalty terms, where α determines how much emphasis each term receives. The optimal α value could be specified by the user or found via hyperparameter tuning.

1.1.3.3 Nearest Neighbor Algorithms

Nearest neighbor algorithms are memory-based methods where an unknown point is classified or estimated using information from other data points close to it (i.e., the nearest neighbors). Nearest neighbor algorithms include *k*-nearest neighbors (*k*-NN), approximate nearest neighbors (Arya et al., 1998), and t-distributed stochastic neighbor embedding (Van der Maaten & Hinton, 2008), among which *k*-NN is the most widely used.

k-Nearest Neighbors: *k*-NN could be used for both classification and regression. When used for classification, *k*-NN classifies an unknown observation through its "nearest neighbor" whose outcomes are known (Altman, 1992; Fix & Hodges, 1951). *k* represents the number of nearest neighbors used to classify the unknown observation. As shown in Figure 1.3, Example 1 (top panel), when $k = 1$, we label the unknown

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

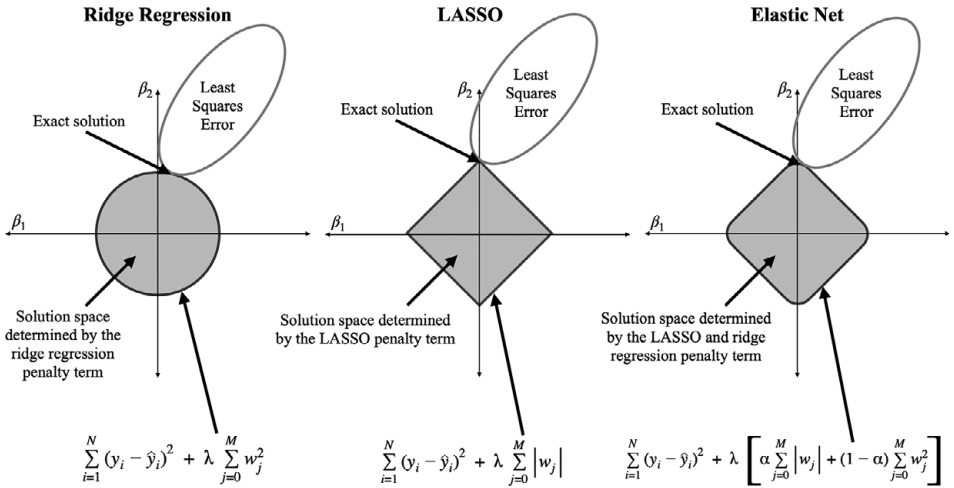

Figure 1.2 Comparison among three regularization algorithms: ridge, LASSO, elastic net

Note. This figure provides a comparison among the three common regularization algorithms. For each algorithm, the penalty term (represented by the black bolded border) constrains the feature weights to a limited solution space (represented by the gray-shaded area). The intersection between the solution space (gray-shaded area) and the least squares error (ellipse) provides a solution that minimizes the cost function of each algorithm (ridge, LASSO, elastic net).

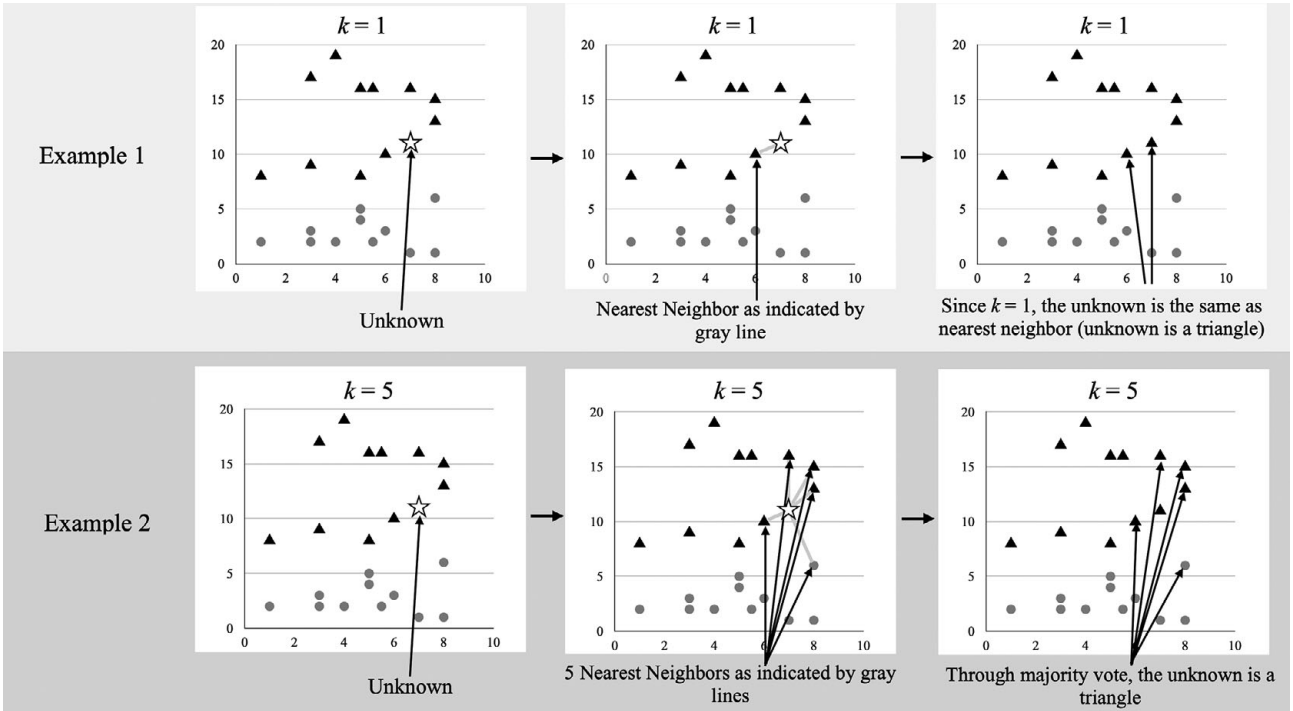

Figure 1.3 Illustration of *k*-nearest neighbors

Note. *k*-NN uses the nearest neighbors to determine how to classify the unknown value (represented by the star). If $k = 1$ (Example 1), the unknown is determined by the data point that is closest to it. As the nearest neighbor is a triangle, the unknown value is also labeled a triangle. If $k = 5$ (Example 2), the classification of the unknown is determined by the five data points closest to it. As four out of the five nearest neighbors are triangles (while one is a circle), the unknown data point is classified as a triangle.

(represented by the star) based on the closest data point, which is a triangle. Thus, the unknown value is also classified as a triangle. If $k > 1$, the classification of the unknown observation is determined by a majority vote from the k nearest neighbors. As shown in Figure 1.3, Example 2 (bottom panel), if $k = 5$, we first identify five nearest neighbors of the unknown point (star). The five nearest neighbors include four triangles and one circle. Thus, through majority vote, the unknown value is labeled as a triangle. For regression models, the predicted outcome of an unknown observation is a weighted average, where the weights are the inverse distance between each neighbor and the unknown observation.

k-NN has a number of advantages. It does not assume linearity, and thus could be used to model different relations flexibly. It is also a memory-based model where no pretraining of the algorithm is required (i.e., "lazy learning"; Bontempi et al., 1999). However, *k*-NN requires observations from all possible feature combinations, and thus is susceptible to the "curse of dimensionality" (Kouiroukidis & Evangelidis, 2011) and requires high computational power (Kotsiantis, 2007). It also assumes all features are equally important, unlike regularization that allows for feature selection or regularization of less important features.

1.1.3.4 Tree-Based Models

Tree-based models are recursive partitioning methods that are nonparametric and highly flexible. The most common tree-based models include decision trees (Morgan & Sonquist, 1963), bagged trees (Breiman, 1996), random forests (Breiman, 2001), and gradient-boosted trees (Friedman, 2001).

Decision Trees: Decision trees are the simplest of the tree-based algorithms and use if-then decision rules (Quinlan, 1986). The algorithm begins by selecting features that account for the most unexplained variance in the data. It then partitions the data based on different values of the features, and this process is repeated until all the observations are classified or estimated. For classification trees, the outcome is the probability (%) of an observation belonging to a certain group; for regression trees, the outcome is a continuous estimate (James et al., 2013). Figure 1.4 provides an example of a decision tree for classification. In the example, Feature 1 is first used to partition the data. The data points that meet Feature 1's cutoff is classified as "A," and the rest is further partitioned using Feature 2. The remaining features are subsequently introduced to the model, and the process is repeated until the classification is complete.

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

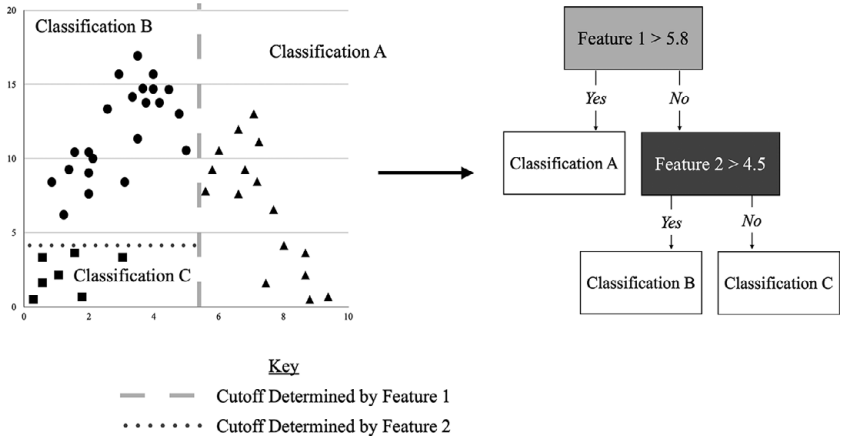

Figure 1.4 Illustration of decision tree

Note. Illustrates a classification tree. Data is split based on the cutoffs (or decision rules) determined by the features. For example, Feature 1 (in light gray) is the feature that accounts for the most unexplained variance in the data and therefore is introduced to the model first. The data points whose Feature 1 value is greater than the 5.8 cutoff (as indicated by the dashed line) are classified as "A" and the rest is further partitioned using Feature 2. The data points whose Feature 2 value is greater than the 4.5 cutoff (as indicated by the dotted line) are classified as "B." The rest of the data points are classified as "C."

Decision trees allow for intuitive interpretation and visualization (as demonstrated in Figure 1.4). They are also flexible in detecting nonlinearity and interactions between features and outcomes (Quinlan, 1986). However, decision trees are limited in that they exhibit high variance (and are thus susceptible to model overfit): the effect of an error in a decision rule affects all of the following splits. Because of this, a second independent sample of observations is often used to prune the decision rules to reduce overfitting and high variance in estimations (Hastie et al., 2009; Myles et al., 2004).

Bagged Trees: Bagged trees address the limitations of decision trees by aggregating the results of multiple decision trees to find one stable outcome (i.e., lower variance; Breiman, 1996). First, the algorithm trains a number of decision trees using independent bootstrapped samples of the training data. These trained trees are then used to estimate the constructs in the test data, and the individual estimates are aggregated to form the final construct estimate. We demonstrate this process in Figure 1.5. First, decision trees are trained using bootstrapped samples of the training data.

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

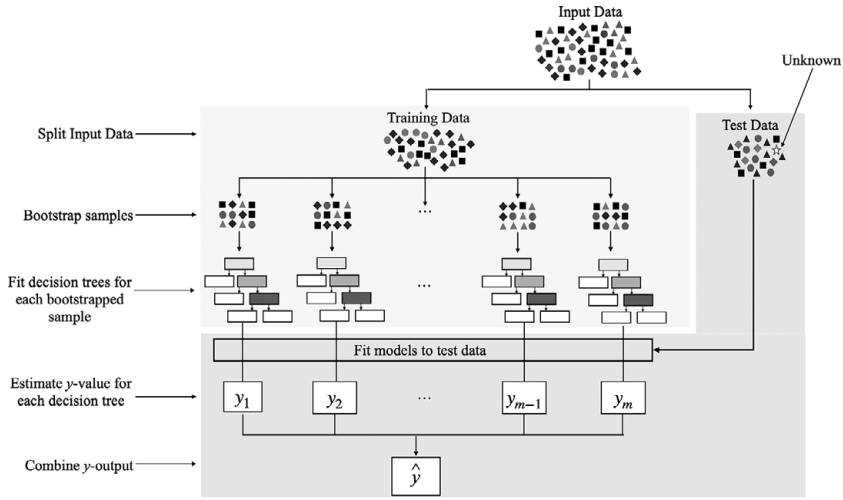

Figure 1.5 Illustration of bagged trees

Note. The aim in this example is to classify or estimate the unknown data point (the star). y is the model estimate and m is the number of decision trees. The light gray area presents the model training process, and the dark gray area presents the process of using the model to estimate constructs in the test set. First, decision trees are trained using bootstrapped samples of the training data. These trained trees are then used to estimate the construct scores in the test set ($y_1, y_2, \ldots, y_m$). The resulting construct estimates from individual decision trees are then aggregated to form the final construct estimate ($\hat{y}$). For regression, the final construct estimate is the average across the decision trees; for classification, the final construct estimate is a majority vote from the classification outcomes of each tree.

These trained trees are then used to estimate the construct in the test set ($y_1, y_2, \ldots, y_m$; indicated by the star). The resulting construct estimates from the individual decision trees are then aggregated to form the final construct estimate (y). For regression, the final construct estimate is the average across the decision trees; for classification, the final construct estimate is a majority vote from the classification outcomes of each tree.

Bagged trees are more recommended than decision trees as they account for the latter's tendency to overfit. However, bagged trees are not without limitations. For example, in bagged trees, the estimated values of the individual decision trees tend to be highly correlated with each other, lacking the diversity needed to reflect the full outcome space and, therefore, may result in low estimation accuracy (Quinlan, 1996).

Random Forests: Random forests address the limitations of bagged trees through feature sampling (Breiman, 2001). Similar to bagged trees,

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

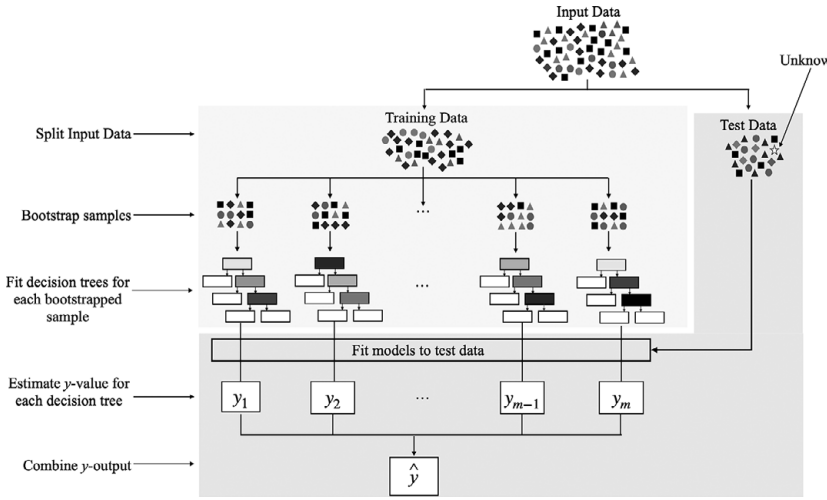

Figure 1.6 Illustration of random forests

Note. y is the model estimate and m is the number of decision trees. Similar to our example of bagged trees (Figure 1.5), the aim is to classify or estimate the unknown data point (the white star). The light gray area presents the model training process, and the dark gray area presents the process of using the model to estimate constructs in the test set. First, decision trees are trained using bootstrapped samples of the training data, where the features included in the trees are *randomly selected*. These trained trees are then used to estimate the construct scores in the test set (y_1, y_2, . . ., y_m). The resulting construct estimates from individual decision trees are then aggregated to form the final construct estimate ($\hat{y}$). For regression, the final construct estimate is the average across the decision trees; for classification, the final construct estimate is a majority vote from the classification outcomes of each tree.

in random forests, multiple decision trees are trained using independent bootstrapped samples, and the trained trees are then used to obtain estimates. The final construct estimates are obtained by aggregating the estimates from the individual decision trees. However, unlike bagged trees that use all features to train the trees, random forests randomly choose the features to train each tree (i.e., feature sampling). By randomly sampling the features, the estimates from the decision trees are less likely to be correlated with each other and thus are more reflective of the outcome space, contributing to improved estimation accuracy (Breiman, 2001). Figure 1.6 illustrates a random forest. Compared to bagged trees (Figure 1.5), in random forests, each decision tree is fitted onto bootstrapped samples of the training data using *random* features, as shown in Figure 1.6, where the various features (boxes) are randomly presented across the decision trees.

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

Gradient-Boosted Trees: While the bagged trees and random forests independently train decision trees, in gradient-boosted trees, the subsequent decision trees improve upon the previous decision trees, further increasing estimation accuracy. The algorithm first trains a decision tree using the full training data. The residuals, or the unexplained variance, from the first tree are then used to train a second decision tree, and this process is repeated until the residuals become smaller than a user-specified threshold. Figure 1.7 shows an illustration of gradient-boosted trees.

Stochastic Gradient-Boosted Trees: Gradient-boosted trees train decision trees using the full training data, and thus could be computationally intense and sensitive to local minima. To address this issue, scholastic gradient-boosted trees (SGBT) randomly sample subsets of the training set to train the individual trees. Each new tree is trained based on previous trees to reduce residuals and improve measurement, especially in areas not well estimated in previous samples. Compared to gradient-boosted trees, SGBT tends to provide more stable and accurate estimates, and is computationally more efficient (Hastie et al., 2009).

1.1.3.5 Support Vector Machines

Support vector machines (SVM) (Drucker et al., 1996) offers a robust method for predicting outcomes with nonlinear relation to the features. SVM does so by using kernel functions to apply mathematical transformations of the feature space to model complex relations. SVM also relies only on a small but critical subset of the training data to guide classification and estimation, and thus is less influenced by outliers. Thus, when there are many features compared to the sample size (i.e., high p to n ratio), SVM tends to perform better compared to regression and tree-based approaches (Kotsiantis, 2007). Finally, SVM uses an "error tube" to allow for a certain level of error in classification and estimation, making it possible to make classification or estimation in cases of complex relations.

1.1.3.6 Neural Networks

Neural networks are inspired by the way the human brain operates. Similar to the way neurons propagate signals, neural networks consist of complex interconnected structures that allow them to discover patterns and recognize underlying relations within a set of data (Hanson & Salamon, 1990). In addition, unlike other ML algorithms that require the data to be

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

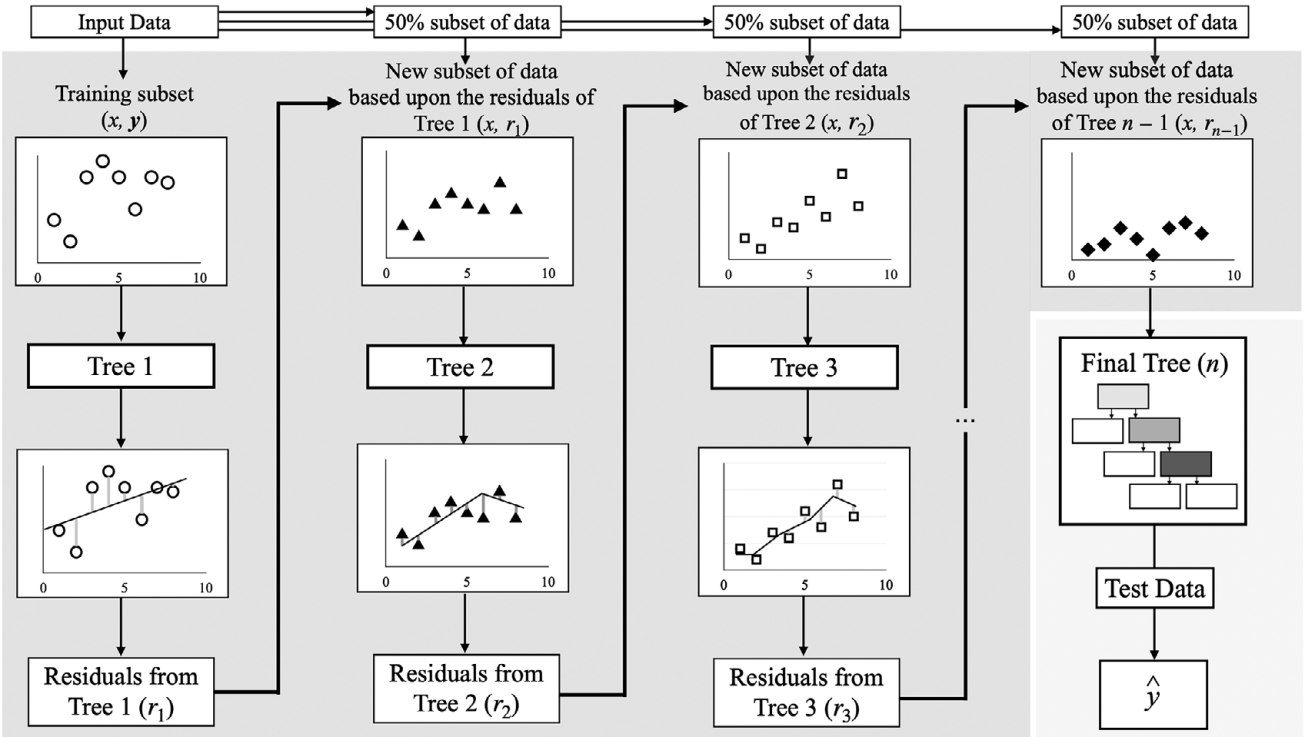

Figure 1.7 Illustration of stochastic gradient-boosted trees

Note. First, a decision tree, Tree 1, is trained with the entire training data (circles) to estimate the outcome, y. The residuals of Tree 1, γ_1, are calculated as the difference between the estimated and observed outcomes (y). Then, a new decision tree, Tree 2, is trained with a randomly selected subset of the training data to estimate the Tree 1 residual, γ_1. This process (where the residuals from the previous tree are used to train a new tree) is repeated over n iterations until the residuals are constant. The last decision tree is the final decision tree and is used to estimate the construct score ($\hat{y}$).

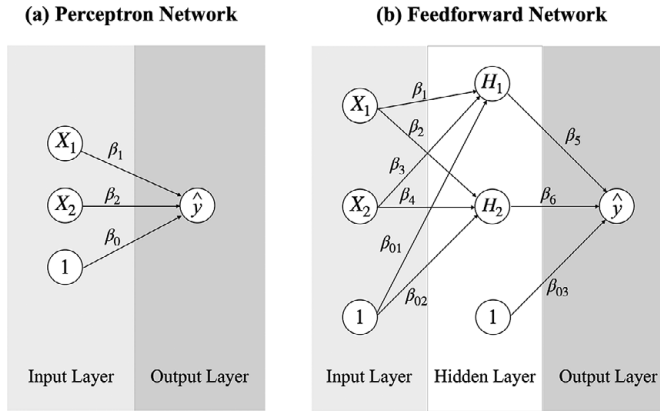

Figure 1.8 Illustration of neural networks: a perceptron neural network and a feedforward network with a single hidden layer

Note. Panel (a) represents the perceptron network, where multiple features connect to one output layer. X represents the input variables (or features). β represents a weight value which the value of the connected variable is multiplied by to obtain the output passed to the receiving node. This weight is adjusted during training to minimize prediction error. $\hat{y}$ represents the estimated construct score which is computed from multiplying, summing, and transforming the values of the connected input variables. Panel (b) represents the feedforward network which adds the additional "hidden layer" between the input variables and the estimate. In this panel, H represents the value computed from multiplying, summing, and transforming the values of the connected input variables.

distilled into important features, neural networks perform feature engineering within the network. These properties make neural networks adept at analyzing unstructured data with ill-defined features, including text, images, and audio.

The simplest form of a neural network is called a perceptron. Perceptrons have a series of input nodes (the features) that connect to a single output node. Figure 1.8a presents a perceptron with two input nodes that correspond to two features: x_1 and x_2. It also includes a constant node; if the constant is 1 (as shown in Figure 1.8a), the perceptron is mathematically equivalent to a linear regression. Each connection has a specific weight (e.g., β_0, β_1, β_2 in Figure 1.8a) that the neural network can modify to better estimate the outcome.

Neural networks typically consist of multiple perceptrons, allowing them to model complex relations. Perceptrons whose outputs are inputs into another perceptron are called "feedforward neural networks" (Bengio, 2009). As demonstrated in Figure 1.8b, feedforward neural networks add an additional layer of neurons between the input and the output, called the

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

hidden layer (in Figure 1.8b, the two nodes in the hidden layer are labeled "H_1" and "H_2"). The hidden layers allow feedforward networks to model complex relations between inputs and outputs (Lu et al., 2017). Networks with many hidden layers are termed "deep neural networks."

The goal of training a neural network is to adjust the weight values in the network so that the transformation of the input features is as close to the desired output as possible. Neural networks are trained in an iterative process. Rather than sending all of the training data at once through the network, the network works on a subset of the training data in each iteration. In each iteration, the weights obtained from the previous iterations are used to estimate outputs in the current subset, and the estimated outputs are compared to the "ground truth." Based on how well the estimated outputs approximated the "ground truth" (i.e., the magnitude of the error), the weights are further updated. After being provided all of the data, the network can revisit the dataset, and continue to adjust its weights, until the errors reach a certain acceptable threshold. This process of using errors in previous weights to inform further weight updates is called "backpropagation." An example of backpropagation is shown in Figure 1.9. The goal of the network is to identify whether the 9 pixel $\times$ 9 pixel image (on the left of the figure) is a diamond or a square. The network includes 81 input features (see "Input Layer"), representing the color (gray vs. black) of the 81 pixels in the image. First, the network generates a set of weights through forward propagation. The network uses this set of weights to identify the image, and the output suggests a 20% probability that the image is a diamond, and an 80% probability that the image is a square. Because the image is a diamond (i.e., the ground truth is 100% probability that the image is a diamond and 0% probability that the image is a square), the current outputs are inaccurate. To improve the accuracy, the network revisits the dataset and continues to adjust its weights through backpropagation. Backpropagation allows the network to eventually approximate the data with high accuracy (Lu et al., 2017).

Neural networks can easily overfit to the training data, especially if the model is overly complex. One can mitigate this problem by limiting the number of times (i.e., epochs) the training data passes through the neural network when finding weights or using ridge or LASSO to regularize the weights. Some analysts mitigate overfitting through "dropout," or temporarily removing a random proportion of nodes. Additionally, analysts can control the learning rate so that changes to the weights are gradual and less influenced by individual data points.

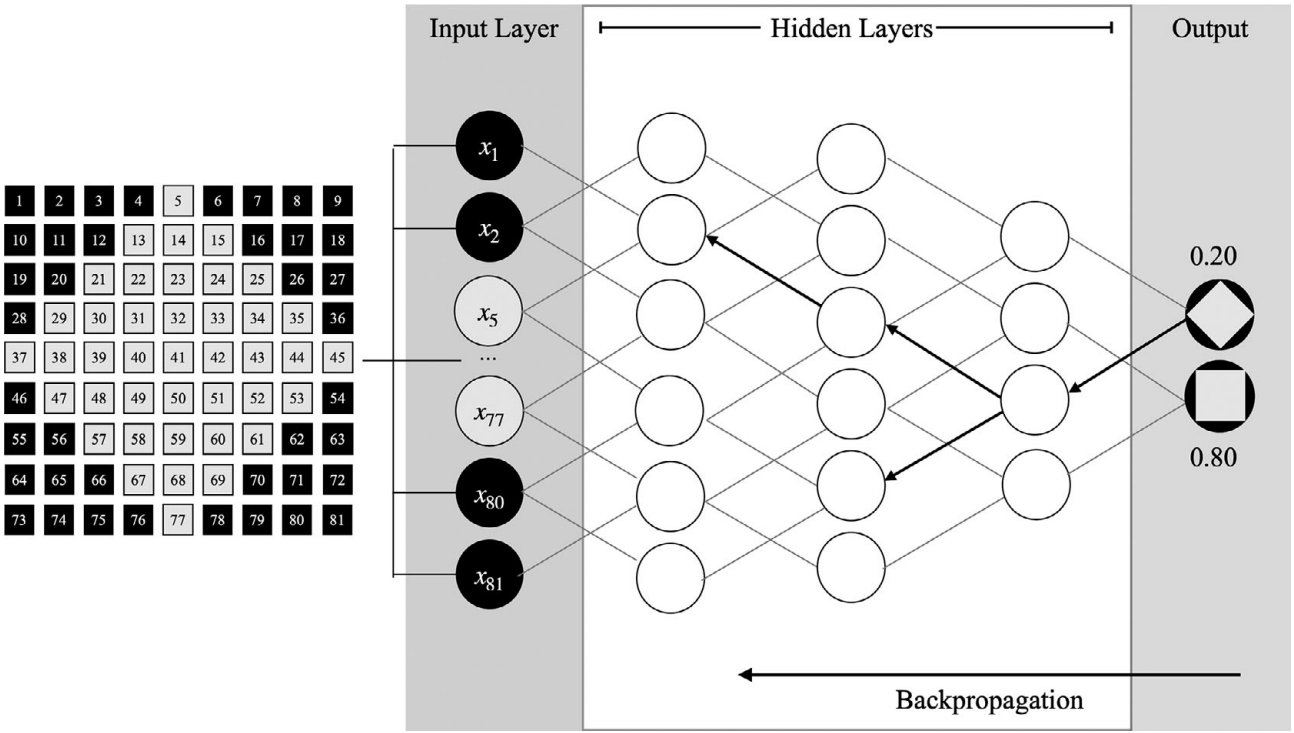

Figure 1.9 Illustration of neural networks: backpropagation

Note. When estimating the shape within the far-left image, data are input into the input layer (pixels) and forward propagated through the hidden layers and to the output. As seen in the output layer, the neural network inaccurately estimated that the shape is likely a square (0.80). To correct this error, the network updates the weights (white circles) in the hidden layers to through back propagation (indicated by the bolded arrows). This process is repeated until the correct shape is estimated the majority of the time.

Example Applications of Supervised Learning Algorithms

The applications of most supervised learning algorithms follow the same procedures described in Section 1.1.1 "General Procedures of Using ML Algorithms in Measurement" (e.g., identify "ground truth," train model, test model). We now briefly describe a number of example applications.

Elastic Net Regression: Hickman, Bosch, et al. (2022) used elastic net regression to measure Big Five personality traits from automated video interviews. The researchers separately trained two kinds of elastic net models, each treating self-reported and interviewer-reported Big Five personality traits as the "ground truth." To train the models, the researchers first extracted verbal, paraverbal, and nonverbal behavior indicators from video clips of individuals responding to interview questions. The behavior indicators were converted to numerical vectors, and elastic net models were trained to obtain personality estimates that approximate the "ground truth."

Random Forest: Speer (2020) used random forest to measure job performance from performance appraisal narratives. Performance appraisal narratives are textual descriptions of employee job performance, typically provided by supervisors; they provide organic descriptions of employees' performance, as illustrated by the supervisor. The researchers first used natural language processing techniques (e.g., *n*-gram scoring) to convert texts into numerical vectors, and then used random forest to obtain job performance estimates from the numerical vectors. Results showed that the job performance estimates converged with human ratings of the textual narratives and demonstrated validity across different samples.

Neural Networks: Ahmad et al. (2020) used deep neural networks to measure personality dark triads from tweets (from Twitter). Three psychiatrists annotated each tweet in the training set to "dark triad" (e.g., psychopath) and "light triad" (e.g., nonpsychopath); these annotations served as the "ground truth" for model training. To train the neural network model, the researchers first used natural language processing techniques (e.g., work embedding) to preprocess the textual tweets into numeric vectors, and then trained neural network models to obtain personality dark-triad estimates from the numeric vectors.

1.2 Recommendations for Using ML Algorithms in Measurement

A number of factors could influence our choice of ML algorithms for measurement. These factors include: (1) the balance between

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

interpretability and estimation accuracy, (2) the missingness present in the training data, and (3) the computational demands of the models.

1.2.1 Interpretability versus Estimation Accuracy

ML models differ in their interpretability: for some models, it is easy to understand how the inputs are transformed into an output value, while that is not the case for others. Models whose learned relations are easy to infer and describe are called "white-box" models. Models that generate complex rules more difficult to understand are referred to as "black-box"[4] models.

1.2.1.1 White-Box Models

Overview of White-Box Models: Linear regression is an example of a "white-box" model, where the intercept and regression coefficients clearly illustrate the relation between the features and the estimates and provide sufficient information to reproduce the estimates. In a regression model, the intercept represents the expected value of the outcome, when the features are 0. The regression coefficients of a feature represent the amount of expected change in the outcome per one-unit increase in that feature, holding all other features constant. Decision trees are also considered white-box models because the process of obtaining an outcome from the features is expressed using intuitive declarative rules (e.g., whether a feature value is greater or smaller than a cutoff). Comprehension of these rules does not require sophisticated mathematical training, making the model clear and easy to interpret.

Benefits of White-Box Models: The benefits to using white-box models include (1) theoretical development, (2) defensibility, and (3) model evaluation. White-box models provide clear processes that illustrate the relations among variables, which could help derive parsimonious explanations and thus improve the theoretical understanding of the variables. Additionally, white-box models offer clarity on the features used, the relative importance of the features, and the relation between the feature and the outcome, allowing one to defend the conclusions of the model. Finally, white-box models are intuitive, facilitating the evaluation of the model coefficients to what would be expected by theory, thus promoting greater confidence and trust in the model (Winkielman et al., 2003).

[4] We restrict this term to opaqueness caused by complexity and not proprietary secrecy (Rudin, 2019).

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

1.2.1.2 *Black-Box Models*

Overview of Black-Box Models: Models with architectures that cannot translate to concise explanations are called black-box models. Random forests, SVM, and deep neural networks are considered "black-box" models. Random forests contain hundreds of trees, each with their own set of features and decision rules. SVM projects the feature onto a multimodal nonlinear surface to create decision boundaries. Neural networks contain dozens of sequential hidden layers that nonlinearly transform the outputs of each prior layer. These models are complex, which can confer many advantages, but at the expense of the interpretability, defensibility, and intuitiveness.

Benefits of Black-Box Models: The benefits of black-box models include (1) potentially higher model accuracy, (2) robustness across different problems, and (3) feature engineering capability. First, compared to white-box models, black-box models have the potential to better capture the relation between features and outcomes because they typically contain many more parameters than their white-box counterparts. Second, black-box models are useful for modeling a wide range of feature-outcome relations, especially when the relation is complex. Black-box models are often useful when one does not yet have an a priori picture of the underlying relation. Third, black-box models can automatically engineer features, allowing the models to capture key higher-order relations. Researchers often seek to generate new feature variables from existing variables, such as by aggregating personality items to form a composite. Models like neural networks and random forests create higher-order representations of variables based on the relations reflected in the data, thus increasing the accuracy of the model.

1.2.2 Data Missingness

Training data commonly consist of missing inputs. For example, participants may forget or refuse to complete a response; they may run out of time or not understand the instructions; and data collection software may experience an error or internet outage. These missing data points could significantly impact the performance of some ML algorithms (Rubin, 1976).

Linear regressions, SVM, and neural networks require the input data to be complete (and without missingness). For those algorithms, researchers are advised to use an imputation approach that approximates the true value of the missing data point (e.g., stochastic model-based imputation) to meet

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

the requirement of the algorithm while preserving information conveyed in the data (Newman, 2014). Decision trees, random forests, and gradient-boosted trees can automatically handle missingness through methods such as surrogate identification (e.g., imputing missing value using complete value from a neighboring/similar case; Batista & Monard, 2002), median imputation (e.g., imputing missing value with the median of the observed data for that column), and more sophisticated sparsity-aware splitting (e.g., building decision trees by assigning missing entries to different split sides and determining which assignment provides the largest maximum gain; Chen & Guestrin, 2016). These algorithms do not require any prior imputation, which is useful for situations where imputation is computationally infeasible. However, these missing data treatments that only impute the single most-likely value for a given set of predictor values tend to demonstrate greater error in reproducing the natural variability within the original data. To more accurately reproduce the variance of the columns, researchers can use stochastic model-based imputation, which applies randomness to the predicted/imputed values. (Newman, 2014).

1.2.3 Computational Resources

Users often have limited computational resources and time for training ML models. Certain models require more advanced computational resources, such as random access memory (RAM), computational processing unit (CPU), and graphics processing unit (GPU).

1.2.3.1 Memory Requirements: RAM

RAM is a computer hardware for storing information during computation. Larger RAM allows for more data and parameters to be processed during model training. If the model training requirement exceeds the available RAM, it may result in an error in the model or the computer may terminate the operation. RAM requirement is determined by the sample size, number of features, and the model's parameterization. Larger sample sizes, number of features, and number of parameters require larger RAM. This is especially likely when the model is complex. For example, a large RAM is necessary for random forests with large numbers of trees, SVM with complex kernels, and neural networks with multiple hidden layers; while small to moderate RAM could generally satisfy the needs of linear regression models. When the model requires large RAM for training, one could consider using batched training (e.g., learning from 50 to 1,000

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

instances at a time via stochastic gradient descent optimization; Zhang, 2004) to make memory usage flexible and reduce the RAM needs. Similarly, bagged methods can minimize RAM usage by modifying the number of features and bootstrap proportions in each iteration.

1.2.3.2 Computation Speed Requirements: CPU

CPU is responsible for carrying out the model training process. A CPU with faster computational speed (i.e., clock rate) can train a model in a shorter amount of time. While insufficient RAM will introduce errors to model training, an underpowered CPU will prevent a model from being trained altogether. Typically, faster CPUs are beneficial for complex models (that require numerous parameters to estimate) and models with large sample size and number of features. Models that are simple and require a few computational steps are the least CPU intensive. Linear regression and *k*-nearest neighbors generally require less CPU, whereas random forests, SVM, and neural networks are more CPU intensive. In cases where the model training requires large CPU capacity, one could consider using parallel processing (Brownlee, 2020; Kuhn, 2019). Parallel processing partitions a computational task into multiple smaller tasks (e.g., constructing a single tree of the random forest) and distributes them to multiple cores on a computer to work on simultaneously, therefore shortening the total time to complete the task.

1.2.3.3 Graphics Computation Requirements: GPU

GPUs are a component of the computer that is specialized at graphical computations (e.g., calculating lighting angles, movement trajectory, and collision). These tasks require linear algebra computations that often take CPUs weeks, months, or even years to complete, while GPUs could perform the computation in parallel, greatly reducing the computation time (Fujimoto, 2008). For example, neural networks typically require GPUs when the features include text, audio, timeseries, and image analysis. GPUs are equipped with their own memory storage functionality, yet just as with RAM, it might not be sufficient for certain applications when the model is complex or the number of features is large. To address this limitation, one might use smaller batch sizes or switch the precision of the calculations to 16-bit instead of the typical 32-bit (akin to rounding a long decimal number). This precision change negligibly changes the model coefficients, but allows the calculations to proceed within memory capacity.

1.2.4 How to Learn More?

There are many readily available resources to help researchers and practitioners apply ML to measurement, which includes click-and-point programs (e.g., IBM SPSS Statistics for Windows, IBM Corp., 2020; Shiny R package, RStudio, 2020), as well as more flexible programming languages – R and Python. Both R and Python are free and open-source programming languages that are widely used for machine learning. Yet, they are distinct in terms of (1) purpose, (2) usability, and (3) flexibility (Krajewski, 2020). First, the two languages' *purpose* is different. R is specifically developed for statistical analysis, while Python is developed for more general programming. If your focus is data analysis or statistical modeling, you may prefer R. However, if your focus is to integrate data analytics and statistical capabilities into a production workflow, you may prefer Python. Second, in terms of *usability*, R is equipped with a wide variety of packages for statistical analyses and measurement (e.g., general linear modeling, item response theory); while Python offers packages focusing on data processing and machine learning (e.g., for natural language processing and computer vision). Third, Python is generally more *flexible* than R. Python codes can easily be integrated into existing software architectures including back-end and cloud architecture, while R lacks such functionalities. Table 1.2 presents a number of useful resources and tutorials are available for implementing machine learning with R and Python.

Table 1.2 *Resources and tutorials for implementing machine learning in R and Python*

Programming language	Official documentation	Tutorial books	Online communities
Python	https://docs.python.org https://python.readthedocs.io	Géron (2019) McKinney (2012) Müller and Guido (2018) Nelson (2020)	*Python Machine Learning Tutorials* *Python Machine Learning*
R	https://cran.r-project.org/manuals.html www.rdocumentation.org	James et al. (2013) Kuhn and Johnson (2013) Wickham and Grolemund (2017)	*R-bloggers* *stat.ethz.ch Mailing Lists*

https://doi.org/10.1017/9781009099813.003 Published online by Cambridge University Press

1.3 Conclusion

The technological advancements in measurement are accompanied by machine learning. The current chapter provided an overview of common ML algorithms used in measurement. ML algorithms enable the use of complex, organic data, and contribute to two key elements of measurement: conceptualization and empirical keying. The current chapter provided recommendations and resources for using ML algorithms for measurement, emphasizing the interpretability and estimation accuracy, and describing best practices for selecting ML algorithms and tools. Recent developments in technology and measurement – many of them highlighted in the edited volume – suggest a promising potential for the future of measurement. Machine learning has a key role to play in the advancement of measurement, and we hope this chapter could help you be equipped with this powerful tool.

REFERENCES

Ahmad, H., Arif, A., Khattak, A. M., Habib, A., Asghar, M. Z., & Shah, B. (2020, January). Applying deep neural networks for predicting dark triad personality trait of online users. In *2020 International Conference on Information Networking (ICOIN)* (pp. 102–105). IEEE.

Akhtar, R., Winsborough, D., Ort, U., Johnson, A., & Chamorro-Premuzic, T. (2018). Detecting the dark side of personality using social media status updates. *Personality and Individual Differences*, *132*, 90–97. https://doi.org/10.1016/j.paid.2018.05.026

Altman, N. S. (1992). An introduction to Kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

Angelov, D. (2020). Top2Vec: Distributed representations of topics. *ArXiv:2008.09470 [Cs, Stat]*. http://arxiv.org/abs/2008.09470

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., & Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, *45*(6), 891–923. https://doi.org/10.1145/293347.293348

Batista, G., & Monard, M.-C. (2002). A study of K-nearest neighbour as an imputation method. In International Conference on Health Information Science.

Bengio, Y. (2009). *Learning deep architectures for AI*. Now Publishers Inc.

Blei, D. M., Jordan, M. I., Griffiths, T. L., & Tenenbaum, J. B. (2003). Hierarchical topic models and the nested Chinese restaurant process. *Proceedings of the 16th International Conference on Neural Information Processing Systems*, 17–24.

Bontempi, G., Birattari, M., & Bersini, H. (1999). Lazy learning for local modelling and control design. *International Journal of Control*, *72*(7–8), 643–658. https://doi.org/10.1080/002071799220830

Braun, M. T., & Kuljanin, G. (2015). Big data and the challenge of construct validity. *Industrial and Organizational Psychology: Perspectives on Science and Practice*, *8*(4), 521–527. https://doi.org/10.1017/iop.2015.77

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

(2001). Random forests. *Machine Learning*, *45*, 5–32. https://doi.org/10.1023/A:1010933404324

Brownlee, J. (2020, September 21). Multi-core machine learning in Python with Scikit-Learn. *Machine Learning Mastery*. https://machinelearningmastery.com/multi-core-machine-learning-in-python/

Calanna, P., Lauriola, M., Saggino, A., Tommasi, M., & Furlan, S. (2020). Using a supervised machine learning algorithm for detecting faking good in a personality self-report. *International Journal of Selection and Assessment*, *28* (2), 176–185. https://doi.org/10.1111/ijsa.12279

Cardaioli, M., Cecconello, S., Conti, M., Pajola, L., & Turrin, F. (2020). Fake news spreaders profiling through behavioural analysis. In *Working notes of CLEF 2020 – Conference and Labs of the Evaluation Forum*, Thessaloniki, Greece, September 22–25, 2020, vol. 2696 of CEUR Workshop Proceedings. CEUR-WS.org

Chaney, A., & Blei, D. (2012, May). Visualizing topic models. In *Proceedings of the International AAAI Conference on Web and Social Media* (Vol. 6, No. 1).

Chekroud, A. M., Gueorguieva, R., Krumholz, H. M., Trivedi, M. H., Krystal, J. H., & McCarthy, G. (2017). Reevaluating the efficacy and predictability of antidepressant treatments: A symptom clustering approach. *JAMA Psychiatry*, *74*(4), 370–378. https://doi.org/10.1001/jamapsychiatry.2017.0025

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).

Costa, P. T., & McCrae, R. R. (1992). Normal personality assessment in clinical practice: The NEO Personality Inventory. *Psychological Assessment*, *4*(1), 5–13. https://doi.org/10.1037/1040-3590.4.1.5

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1996). Support vector regression machines. *Advances in Neural Information Processing Systems*, 155–161.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*(2), 407–499. https://doi.org/10.1214/009053604000000067

Fix, E., & Hodges, J. (1951). Discriminatory analysis – nonparametric discrimination: Consistency properties. Technical Report 21-49-004,4, U.S. Air Force, School of Aviation Medicine, Randolph Field, TX.

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/aos/1013203451

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, *38*(4), 367–378. https://doi.org/10.1016/S0167–9473(01)00065-2

Fujimoto, N. (2008). Faster matrix-vector multiplication on GeForce 8800GTX. *2008 IEEE International Symposium on Parallel and Distributed Processing*, 1–8. https://doi.org/10.1109/IPDPS.2008.4536350

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn & TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc.

Gradus, J. L., Rosellini, A. J., Horváth-Puhó, E., Street, A. E., Galatzer-Levy, I., Jiang, T., Lash, T. L., & Sørensen, H. T. (2020). Prediction of sex-specific suicide risk using machine learning and single-payer health care registry data from Denmark. *JAMA Psychiatry*, *77*(1), 25–34. https://doi.org/10.1001/jamapsychiatry.2019.2905

Groves, R. M., Fowler Jr, F. J., Couper, M. P., Lepkowski, J. M., Singer, E., & Tourangeau, R. (2011). *Survey methodology* (Vol. 561). John Wiley & Sons.

Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(10), 993–1001.

Hassanien, A. E., Kilany, M., Houssein, E. H., & AlQaheri, H. (2018). Intelligent human emotion recognition based on elephant herding optimization tuned support vector regression. *Biomedical Signal Processing and Control*, *45*, 182–191. https://doi.org/10.1016/j.bspc.2018.05.039

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media.

Hickman, L., Bosch, N., Ng, V., Saef, R., Tay, L., & Woo, S. E. (2022). Automated video interview personality assessments: Reliability, validity, and generalizability investigations. *Journal of Applied Psychology*, *107*(8), 1323–1351. https://doi.org/10.1037/apl0000695

Hickman, L., Song, Q. C., & Woo, S. E. (2022). Evaluating data. In K. R. Murphy (Ed.), *Data, methods and theory in the organizational sciences* (pp. 98–123). Society of Industrial and Organizational Psychology Organizational Frontiers Series. Routledge.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67. https://doi.org/10.1080/00401706.1970.10488634

IBM Corp. (2020). *IBM SPSS statistics for Windows, version 27.0*. IBM Corp.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (Eds.). (2013). *An introduction to statistical learning: With applications in R*. Springer.

Jiang, T., Gradus, J. L., & Rosellini, A. J. (2020). Supervised machine learning: A brief primer. *Behavior Therapy*, *51*(5), 675–687. https://doi.org/10.1016/j.beth.2020.05.002

Jung, Y., & Suh, Y. (2019). Mining the voice of employees: A text mining approach to identifying and analyzing job satisfaction factors from online employee reviews. *Decision Support Systems*, *123*, 113074. https://doi.org/10.1016/j.dss.2019.113074

Kenney, J. F., & Keeping, E. S. (1962). Linear regression and correlation. *Mathematics of Statistics*, *1*, 252–285.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica (Slovenia)*, *31*(3), 249–268.

Kouiroukidis, N., & Evangelidis, G. (2011). The effects of dimensionality curse in high dimensional kNN search. *2011 15th Panhellenic Conference on Informatics*, 41–45. https://doi.org/10.1109/PCI.2011.45

Krajewski, R. (2020, November 26). *Python vs R: What language is better for data science projects?* Ideamotive. https://www.ideamotive.co/blog/python-vs-r-what-language-is-better-for-data-science-projects.

Kuhn, M. (2019). Parallel processing. In *The caret package*. https://topepo.github.io/caret/parallel-processing.html

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. Springer.

Lu, Z., Pu, H., Wang, F., Hu, Z., & Wang, L. (2017). The expressive power of neural networks: A view from the wdth. *ArXiv:1709.02540 [Cs]*. http://arxiv.org/abs/1709.02540

MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, *1*(14), 281–297.

Matusik, J. G., Heidl, R., Hollenbeck, J. R., Yu, A., Lee, H. W., & Howe, M. (2019). Wearable bluetooth sensors for capturing relational variables and temporal variability in relationships: A construct validation study. *Journal of Applied Psychology*, *104*(3), 357–387. https://doi.org/10.1037/apl0000334

McKinney, W. (2012). *Python for data analysis*. O'Reilly Media.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Morgan, J. N., & Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American Statistical Association*, *58*, 415–434.

Müller, A. C., & Guido, S. (2018). *Introduction to machine learning with Python: A guide for data scientists*. O'Reilly Media.

Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., & Brown, S. D. (2004). An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, *18*(6), 275–285. https://doi.org/10.1002/cem.873

Nelson, D. (2020). *Data visualization in Python*. StackAbuse.

Newman, D. A. (2014). Missing data: Five practical guidelines. *Organizational Research Methods*, *17*(4), 372–411. https://doi.org/10.1177/1094428114548590

Park, G., Schwartz, H. A., Eichstaedt, J. C., Kern, M. L., Kosinski, M., Stillwell, D. J., Ungar, L. H., & Seligman, M. E. P. (2015). Automatic personality assessment through social media language. *Journal of Personality and Social Psychology*, *108*(6), 934–952. https://doi.org/10.1037/pspp0000020

Polyak, S. T., von Davier, A. A., & Peterschmidt, K. (2017). Computational psychometrics for the measurement of collaborative problem solving skills. *Frontiers in Psychology*, *8*. https://doi.org/10.3389/fpsyg.2017.02029

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

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106. https://doi.org/10.1007/BF00116251

(1996). Bagging, boosting, and C4.5. *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 725–730.

Roberts, M. E., Stewart, B. M., Tingley, D., Lucas, C., Leder-Luis, J., Gadarian, S. K., Albertson, B., & Rand, D. G. (2014). Structural topic models for open-ended survey responses. *American Journal of Political Science*, *58*(4), 1064–1082. https://doi.org/10.1111/ajps.12103

RStudio (2020). Learn Shiny. https://shiny.rstudio.com/tutorial/

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, *63*(3), 581–592. https://doi.org/10.1093/biomet/63.3.581

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215. https://doi.org/10.1038/s42256–019-0048-x

Salehzadeh, R. (2017). Which types of leadership styles do followers prefer? A decision tree approach. *International Journal of Educational Management*, *31*(7), 865–877. https://doi.org/10.1108/IJEM-04-2016-0079

Speer, A. B. (2020). Scoring dimension-level job performance from narrative comments: Validity and generalizability when using natural language processing. *Organizational Research Methods*, 10944281209308l. https://doi.org/10.1177/1094428120930815

Steinhaus, H. (1956). Sur la division des corps matériels en parties. *Bulletin L'Académie Polonaise des Science*, *1*(804), 801–804.

Tay, L., Woo, S. E., Hickman, L., & Saef, R. M. (2020). Psychometric and validity issues in machine learning approaches to personality assessment: A focus on social media text mining. *European Journal of Personality*, *34*(5), 826–844. https://doi.org/10.1002/per.2290

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11). http://www.jmlr.org/papers/v9/vandermaaten08a.html

Wang, W., Hernandez, I., Newman, D. A., He, J., & Bian, J. (2016). Twitter analysis: Studying US weekly trends in work stress and emotion. *Applied Psychology*, *65*(2), 355–378. https://doi.org/10.1111/apps.12065

Wickham, H., & Grolemund, G. (2017). *R for data science: Import, tidy, transform, visualize, and model data*. O'Reilly Media.

Winkielman, P., Schwarz, N., Fazendeiro, T. A., & Reber, R. (2003). The hedonic marking of processing fluency: Implications for evaluative judgment. In J. Musch & K. C. Klauer (Eds.), *The psychology of evaluation: Affective processes in cognition and emotion* (pp. 189–217). Lawrence Erlbaum Associates Publishers.

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

Youyou, W., Kosinski, M., & Stillwell, D. (2015). Computer-based personality judgments are more accurate than those made by humans. *Proceedings of the National Academy of Sciences*, *112*(4), 1036–1040. https://doi.org/10.1073/pnas.1418680112

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. *ICML 2004: Proceedings of The Twenty-first International Conference on Machine Learning* (pp. 919–926). Omnipress. https://doi.org/10.1145/1015330.1015332

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