

**Authoring Guide
for
Cognitive Tutorials for Artificial Intelligence
Purposes and Methods**

Shane T. Mueller
Sarah (Yin-Yin) Tan
Anne Linja
Michigan Technological University

Gary Klein
MacroCognition, LLC

Robert R. Hoffman
Institute for Human and Machine Cognition

This material is approved for public release. Distribution is unlimited. This material is based on research sponsored by the Air Force Research Lab (AFRL) under agreement number FA8650-17-2-7711. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL or the U.S. Government.

Cite as:

Mueller, S.T., Tan, S., Linja, A., Klein, G. and Hoffman, R.R. (2021). "Authoring Guide for Cognitive Tutorials for Artificial Intelligence Purposes and Methods." Technical Report, DARPA Explainable AI Program.



Abstract

The Cognitive Tutorial concept is based on the view that the genuine cognitive challenges to forming functional and accurate mental models of AI systems can be formalized, documented, and "trained in." Its purpose is to serve as a means of global explanation of an AI or machine learning system. A Cognitive Tutorial is created specifically to accelerate proficiency at learning to use intelligent software tools. Therefore, it would be a valuable addition to any "toolkit" for ensuring that intelligent systems are explainable, are adequately explained to users, and the users are satisfied with their understanding of the system. This Report describes the procedures for creating a Cognitive Tutorial, the modules that comprise a Cognitive Tutorial, and example Cognitive Tutorials applied to two AI systems.

Outline

1. Background and Motivation	3
2. Authoring a Cognitive Tutorial	4
3. Learning Modules/Lesson types for a Cognitive Tutorial	10
4. Discussion	23
5. Conclusion	25
Appendix A: Sample Cognitive Tutorial Authoring Example	26
Appendix B: Example Modules	31
Acknowledgement and Disclaimer	43
References	44

1. Background and Motivation

Algorithms for explainable AI (XAI) have been a recent focus of extensive research. There are a number of motivations for this research, including satisfying legal regulations, helping developers find errors in their AI systems, helping users understand why a particular action or decision was made by an AI system, and helping users understand how a system works. Most algorithms for XAI focus on *justification*---providing information that serves as an explanation for *why* a particular action was taken. Justification is a form of *local* explanation, and it is possible that a series of these local explanations help satisfy some of the more *global* goals of explanation---understanding how a system works in general. However, although local explanation algorithms may help justify and persuade a user about the correctness of a particular decision, they may not be efficient at developing an overall understanding of the system. The critical elements of understanding might include knowledge about the situations in which it can be trusted (or should be distrusted), the relative accuracy and cost of using the system versus other approaches (such as a human), particular workarounds for its limitations, and best practices for using it within any application context.

Many of these issues are not addressed by most XAI algorithms, although non-algorithmic approaches are available (see Mueller et al., under review). One approach we find particularly promising is the development of *Cognitive Tutorials for AI (CTAI)*. By Cognitive Tutorial, we mean, in general, an approach for training users about the cognitively-challenging aspects of an AI system. This notion builds on previous work in which we outlined the development of an *Experiential User Guide* (Mueller, Klein, & Burns, 2009; Mueller & Klein, 2011)--methods for using experiential training to help users understand complex software tools. Problems that people face with intelligent software tools can go beyond those caused by poor interface design (which can also afflict such tools), and beyond limitations in understanding the simple components of a tool (which could be alleviated with proper instruction). These problems may be fundamentally cognitive, and likely stem from people having inadequate or improper mental model of how the tool works (Koopman & Hoffman, 2003).

A user may have improper expectations about what the intelligent system is doing, and there will often be a need to provide training that helps the user understand the workings of the software, to some point of performance and also satisfaction. Problems that the user encounters in the use of an intelligent system can often be overcome by trial-and-error experience, but that can drag out unnecessarily, and can also result in unnecessary kludges and workarounds, and mis-calibrated mental models, not to mention frustration-based rejections of AI support systems. Problems that a user encounters are often countered by communication among a community of users who passes down information to other users. But this too often happens by circumstance.

The CTAI concept is based on the view that the genuine cognitive challenges to forming functional and accurate mental models can be formalized, documented, and "trained in." A CTAI is created specifically to accelerate proficiency at learning to use intelligent software tools. It exposes a learner to many of the experiences that an expert will have had over a period of weeks or months of use, allowing the learner to experience the strengths and weaknesses of the tool. It is intended for software tools that perform non-trivial or intelligent functions, and can ultimately help both novice and experienced users gain new and better understanding of their tools.

From our perspective, if developers are serious about having an AI system gain adoption, a CTAI supplement is almost a requirement. Why go through the effort and expense and then issue the system without the necessary preparation?

Consequently, this document provides justification and guidance for developing cognitive tutorials for AI. The process we envision is one that jointly relies on methods from the fields of human factors, training, pedagogy, and AI, and will typically involve a team that uses a variety of approaches for identifying the most important tutorial lessons a particular user group requires, and then develops training materials (possibly including static text, imagery, video, and dynamic/interactive demonstrations) that help support the learning objectives. The goal is to shortcut the laborious process of learning about a system through use alone, and provide critical examples that help identify the strengths, weaknesses, and boundaries of an AI system.

2. Authoring a Cognitive Tutorial

This document provides the background, methodology, and justification for different lessons that might make up a cognitive tutorial. We also include a number of detailed examples of data collection approaches for developing a cognitive tutorial, and example lessons we have developed for several machine learning and clustering systems. We begin with a discussion of the aspects of AI systems that should be considered when creating a cognitive tutorial.

Areas of AI that may require cognitive tutorials

Mueller et al (2009) identified four aspects of intelligent software on which to focus tutorials:

(1) **The data requirements of the tool.** Most AI systems, regardless of their underlying architecture, make use of extensive training data, and also require similar data to use for analysis of individual cases. For example, it is important to understand whether an image classifier trained to classify common plants was trained on images of plants outside North America, or with images at a variety of times of plant maturity, etc.

(2) **Representation and modeling mechanisms used by the tool.** For complex deep networks, the modeling mechanisms are difficult to understand and at times important limitations of the system. For other systems that use ontologies, hand-built knowledge structures, or more model-based representations, these are also critical for interpreting output. For example, Zeiler & Fergus (2014) showed how specific internal layers of a neural network were highly sensitive to the faces of specific dog breeds, suggesting that the face pattern may be an important feature used to distinguish dogs.

(3) **Underlying computations and algorithms used by the tool.** Sometimes, the particular logic and mechanisms within an algorithm are critical to understand--but this depends on the context and goals of the user. For example, many image classifiers provide output activation levels for each of the most likely labels, and use a hierarchical classification system to allow the system to identify an image of a dog as a mammal, a dog, and a dachshund. The algorithms by which these systems assign activation to these different hierarchical levels are opaque, and so it can confuse a user, and a tutorial may help the user interpret these hierarchies appropriately.

(4) Output, display, and visualization provided by the tool. Similarly, image classifiers provide a list of labels with activation levels, but the meaning and significance of the activation levels is not often explained. Users may interpret the values (which are bound between 0 and 1.0) as the probability of the classifier being correct (even though they are not). A tutorial may help a user understand this value and how to interpret it.

Example systems and tutorial aspects

AI systems cover a large range of algorithms and functions. Table 1 provides some examples of aspects that might be well supported by a cognitive tutorial system.

Table 1. Aspects of typical AI systems and candidate learning objectives for these systems.

System	Aspect	Example tutorial learning objective
Image Classifier	Data requirements	Breadth, number, and source of images it is trained on.
Music Recommender system	Representation	Is a song represented in terms of its audio form (deep analysis of tempo, key, etc.); genre labels, or based on other songs liked by listeners (collaborative filtering)?
Autonomous game player	Algorithm	Does the algorithm consider opponents/enemies, or has it learned a successful blind strategy?
Autonomous driver	Data requirements	Is the driver trained specifically on particular roadways?
Video annotation system	Data	Is the system trained on speakers with Russian accents?
Language translator	Representation	How strongly does the word classification algorithm consider the meaning and grammar of surrounding words to identify a token?
Credit report algorithm	Output	How is a credit score used to deny loans?
Smart GPS router	Output	How much of a delay does a red line indicate on the current route?
Smart GPS Router	Data sources	How current are data about traffic delay?

Method/Procedures for Creating a Cognitive Tutorial

Although many training programs are developed in an ad hoc manner, we advocate that a useful cognitive tutorial should be a result of a systematic analysis of a system, its users, and the goals of its use. Even if a tutorial is initiated because of a particular problem that has been identified, this may in fact stem from other more fundamental aspects of the AI system, and so this analysis may uncover other issues that need tutoring. The recommended process begins by collecting data regarding the system and its users. This data is used to identify candidate learning objectives for the tutorials. In our experience, systematic analysis leads to many more learning objectives than can reasonably be implemented as part of a tutorial, and so the next step is to identify a particular sequence of learning objectives that best meet the goals of the overall tutorial. Finally, the tutorials need to be implemented as lessons, and we provide a number of active-learning modules which are reasonable approaches to implementing these learning objectives.

One important element of planning is to identify the appropriate composition of the team creating the tutorials. Several distinct skills are probably necessary:

- A team member with the ability to understand and explore the algorithm or system being developed. This may be a subject-matter expert who serves in an advisory role, or an experienced computer scientist or data scientist familiar with the methods being studied who can play a more central role on the team.
- Members experienced in qualitative interview-based data collection methods for pedagogical and training applications (including expertise in cognitive task analysis).
- Experience in instructional systems design, web design, or video to implement interactive training modules

Multiple roles may be played by a single researcher, but it is normally more feasible to form a team with a set of commentary specialties.

Data Collection

We refer to the variety of activities by which a cognitive tutorial developer gathers requirements and identifies learning objectives as data collection. This may involve analyzing existing documents and artifacts (help forums, third-party training), or may involve interactions with system developers or existing or prospective users. The specific data collection process depends on whether the cognitive tutorial development team has direct access to the tool (some embedded or classified systems may not be available for exploration), whether both expert and non-expert users are available for interviewing (for tools that are in development, neither may be), whether system developers and training personnel will be available, whether existing models or analysis output can be obtained, and whether a user community exists. Consequently, *the first stage in developing a CT is to identify the gaps in understanding a user of the system is likely to encounter*. This data collection has two major goals: To identify learning objectives, and to identify vignettes, stories, and examples for the cognitive tutorial.

Sometimes, substantial information can be learned through examining existing artifacts (see Table 2). We have normally used a variety of these sources to help identify typical problems novices have, and sometimes to identify the proper or correct solutions. These can, of course, be misleading or biased, and so should be examined carefully and evaluated in comparison to other means of data gathering. Furthermore, areas that already have training materials developed may give a good idea for where novice users have problems, but may not make good candidates for a cognitive tutorial, because the training already exists and no new tutorials are necessary.

Table 2. Non-human subjects data collection sources.

Data Collection Method/Source	Advantages and Cautions
Web forums or email listservs	Provides good sense of problems users are having; may be biased toward advanced users; will only exist for relatively mature tools with users supported by organizations.
Social Q&A (e.g., StackExchange)	Upvoted questions and answers may help identify important learning objectives, and multiple feasible answers; will only exist for tools with large user groups.
FAQ documents and bug databases	Provides a good starting point for well-defined software

	systems; will tend to focus on low-level elements of software rather than AI functions.
Official user guides/documentation	Can provide a reasonable set of learning objectives for areas where user guides are incomplete; may help exclude lessons if they have been well-covered by official documentation.
Third-party tutorials, textbook chapters, video and web tutorials	Examining common tutorials can help identify missing pieces and misconceptions users may have. In our experience, these often focus on a minimal correct example, rather than showing how things might go wrong.
User-generated artifacts (models, output, etc.)	If the tool supports saving user-AI sessions, output, or models, it can help identify where errors are made. Many systems will not support such artifacts.
Similar tools/systems	In novel or in-development systems, information might come from other similar systems or users/developers of other systems that use similar algorithms or solve similar problems with other algorithms.

We also advocate collecting data using standard human factors research methods (cognitive task analysis interviews, think-aloud protocols, user surveys, etc.; Crandall, Klein, & Hoffman, 2006). Depending on the purpose, this may require oversight from a human subjects protection board (such as an institutional review board). Although this may involve substantial additional expertise and cost, in our experience the learning objectives and the lessons we create are often heavily influenced by human subjects data collection. Some of the typical sources for human subjects data collection are shown in Table 3. Here, we describe the basic source of data collection, but not the method. For any source, a variety of data collection methods might be used.

Table 3. Human subjects data collection approaches.

Data Collection Source	Advantages and Cautions
Novice users first exposure session	Helps identify novice mental models; novices may fail because of interface or other things not well supported by cognitive tutorial.
Knowledgeable user sessions	Watching users attempt specific assigned tasks may help identify problems and common errors.
Wizard-of-Oz sessions or interpretation of pre-generated results.	For systems that are envisioned, a Wizard-of-Oz method can be used in which a user interacts with a system controlled by a human. Similarly, example results can be given and a potential user can discuss or interpret the results without having access to the system.
Experienced user interviews	Experienced user interviews can help identify workarounds, 'correct' interpretations, and use patterns. However, these users will tend to avoid actions that cause problems with the software.
Developer interviews	Developers can give insights into how a system really works. It may also reveal whether interface and output elements are principled in design.
Trainer interviews	If available, users who have formally or informally trained others can provide many examples of mistakes and misconceptions.
Subject matter experts unfamiliar with AI system	If the system is focused on a domain of expertise (e.g., a particular game, image classification of a specific domain), experts in that domain can provide insight into how they work, and problems they have that might be faced by the AI system, and how current/older systems succeed and fail.
Questionnaires or surveys	Although these can be blunt tools, they can provide a basic thermometer of

	use patterns and common complaints.
First-hand learning/training	If a cognitive tutorial developer can receive informal or existing training in a system, careful note-taking during learning can help identify gaps and misunderstandings.
Other stakeholders	Other stakeholders, such as managers or funders, may help understand high-level goals of a system, common problems faced by many users, or priorities for training.

Identify Potential Learning Objectives

Based on the data collected, the CT developer must next identify candidate learning objectives. At this stage, we recommend this begins by identifying gaps, mistakes, workarounds, errors, and other specific problems encountered, without attempting to create concrete learning objectives. Typical pedagogical advice for identifying learning objectives is that they should be concrete, observable, and measurable. It can involve substantial effort to create actionable learning objectives at this stage, and we have found it more helpful to identify a large number of potential objectives without going through the work of making each one concrete. The practical reason for this is that it can be easy to generate dozens of possible learning objectives that never get selected for developing cognitive training, and it would be wasted effort to make concrete objectives prior to this selection.

Creating good learning objectives that can be translated into tutorial modules is challenging. Thus, our recommendation is to initially identify basic problems/challenges observed during data collection, organized by the four basic areas of tool function (see Figure 1 for an example). Here, we have identified the source of the problem (for later reference) and potential ways to train the user, but have not yet articulated specific learning objectives.

Problem/Challenge	On-line Source Format	Possible Solution
What type of system/tool is used? What is the targeted system?	R website	R Shiny
How to use the system/tool? How to initiate the clustering?	Observation of "First 20 Minutes"	
How the system represents information?		
Identify the data sources	Data analysis	personality online pre-screening questionnaires
How to tell if data is "clustered" enough for clustering algorithms?	stats.stackexchange.com forum	use the Gap statistics. Basically, the idea is to compute a goodness
Normalization of network data (clustering algorithms)	stats.stackexchange.com forum	designed to work on continuous variables, no sense for IP address
K-means clustering scaling	stats.stackexchange.com forum	
Clustering a dense dataset	stats.stackexchange.com forum	select the method and the final clustering solution by which possible
How would PCA help with a k-means clustering analysis? (When should I use PCA?)	stats.stackexchange.com forum	1. Doing PCA before clustering analysis is also useful for dimensionality
Data Preparation for Cluster Analysis	stats.stackexchange.com forum	data normalization and removing correlation among data are often
How the tool work, and how it actually does work?	Algorithm analysis	R Shiny source code
When would I use EM instead of k-means?	stats.stackexchange.com forum	there will be uncertainty about your cluster assignments so it is ideal
The problem of how to choose the number of clusters	Book (The Element of Statistical Learning: Theory and Applications)	Kmeans clustering is a top-down procedure, while other clustering algorithms
Partitioning clustering	Book p.229 (Encyclopedia of Machine Learning)	K-means is the most widely used clustering algorithm. It constructs
Are there cases where there is no optimal k in k-means?	stats.stackexchange.com forum	there is no cluster structure in the data. However, clustering in very
K-means clustering has shortcomings in breast cancer clustering	Book p.514 (The Element of Statistical Learning)	1. it does not give a linear ordering of objects within a cluster; 2. as
K-means will not work well when the clusters are non-convex, spherical	Book p.544 (The Element of Statistical Learning)	K-means use a spherical or elliptical metric to group data points; spherical
When clusters are stretched-out banks, the k-means algorithm	Book p.21 (Mining for Strategic Competitive Intelligence)	density-based clustering approaches (see subsequent paragraphs)
The limitation of a traditional K-means algorithm (unable to cluster	Book p.63 (Network Intrusion Detection using Data Mining)	It is observed that the advantage of SAE (Stacked Auto-Encoder)
the steps for K-means clustering	Book p.85 (Natural Computing for Unsupervised Learning)	1. Select K points as initial centroids.
The huge advantage of k-monoids over k-means	Book p.22 (Mining for Strategic Competitive Intelligence)	the first is less susceptible to outliers than the latter, as the cluster
The huge advantage of density-based clustering schemes (e.g., DBSCAN)	Book p.25 (Mining for Strategic Competitive Intelligence)	clusters do not necessarily have to be cloud-like in order to be dense
Less desirable properties of K-Means: (a) the initial setting,	Book p.224 (Core Concepts in Data Analysis)	(a) the initial setting, i.e. the number of clusters K and initial position
Can the centroids be incorrect even if there is convergence?	CodeCademy Q&A	Yes, absolutely. K-Means clustering finishes once the centroids no
How to visualize K-means clusters in 3D?	YouTube Video	Visualizing K-means algorithm in 3D
Real-time simulation of the K-means clustering algorithm	YouTube Video	K-Means Clustering Example, using different values for n and k
How to produce a pretty plot of the results of k-means clustering?	stats.stackexchange.com forum	library(cluster); library(fpc)
Insufficient interpretation aids	Book p.225 (Core Concepts in Data Analysis)	

Figure 1. Screenshot of spreadsheet used to identify basic learning challenges for k-means clustering.

Appendix A includes several additional similar tables. Altogether, we identified more than 100 potential learning objectives for the k-means clustering algorithm--more than could feasibly be turned into actual tutorials--and so we generally only write specific learning objectives for these challenges once a small set has been identified for the final tutorials.

Furthermore, it may be useful to plan the data collection in several rounds from general (published documents, tutorials, and Q&A sources) to specific (interviews with novice and expert users) , so that the CT team gains knowledge and is better able to focus interviews on critical topics.

Prioritize and concretize learning objectives

To create a final set of tutorials, the learning objectives (LOs) need to be prioritized and made concrete. The prioritization should consider:

- Who the tutorials are for (experience and usage goals)
- What the training will be used for
- Extent of other training available
- Resources available to create training tutorials
- Amount of time available for training
- Importance of particular learning objectives
- Practicality of implementing tutorials.

In our experience, this stage generally represents a massive winnowing of dozens of potential learning goals to a handful of concrete learning objectives. We expect that all stakeholders (sponsors, developers, and users) can help with this process, because they may have good insight into what is important and what is not. As a practical matter, the CT will need to balance basic introductory training with modules that enable deeper understanding of the AI. At this point, it is also useful to consult stakeholders and re-examine data sources that provide basic training to see where existing training resources focus, so that effort can be deployed on learning higher-level concepts.

Although most of these concerns in the list are self-explanatory, one less-obvious concern centers on the practicality of implementation. For example, sometimes tutorial builders or users do not have access to the actual system, and must rely on videos or screenshots of the system. Sometimes, a learning objective is not easy to train without direct access to the system, unless it is simulated via other software, which might make the lesson impractical.

As the final objectives are identified, they should be written down as concrete learning objectives that are measurable and observable. When written appropriately, these can later be used for assessing the effectiveness of the tutorial. Table 4 shows some example good and bad LOs, for a case in which users are misinterpreting the output of an AI system (representing activation) as a probability.

Table 4. Example learning objectives including strengths and weaknesses.

Statement of Learning Objective	Assessment	Rationale
Learner should understand system output	bad	Too vague
Learner should be able to interpret output values	bad	Not measurable/poorly defined
Learner should be able to describe three ways in which system output is different from probability	good	Specific, measurable
Learner should be able to distinguish between an accurate and inaccurate description of system output	good	Specific, measurable, relevant to system performance

Map learning objectives on to training modules and implement training

The next step is to identify the best ways to implement tutorial lessons for particular learning objectives. A central concept for the *cognitive tutorial* is to help a user understand the cognitive functions of an AI system. Because cognitive functions are complex, contextual, and involve computation, training cannot normally just involve teaching about the mechanisms of the underlying algorithms. Rather, experiential training in *how the system works in a domain* is central to the CT concept. Next, in Section 3, we describe experiential learning approaches and other learning approaches that are especially well-suited for implementing cognitive tutorials for AI. Appendix B shows and describes examples of these approaches implemented for several distinct AI and ML systems.

3. Learning Modules/Lesson types for a Cognitive Tutorial

Once a set of learning objectives has been identified, implementing tutorial lessons can follow. Mueller & Klein (2009) identified several kinds of experiential learning modules they argued are especially well-suited for teaching about intelligent software tools. We describe those here along with a number of other methods well-suited for understanding and explaining AI. The goal of the tutorial is to help users understand the cognitively challenging aspects of an AI system. We suggest that when possible, example-based, interactive, and experiential learning methods should be used to support this. Table 5 describes different types of tutorial modules we have explored for this purpose. More examples of these tutorial types appear in Appendix B.

Table 5. Examples of modules that can be used for training AI concepts using cognitive tutorials.

Module	Description
A 'walkthrough'	Basic examples of how to use the tool
Forced-choice scenarios	Learner chooses between proper/improper use or interpretation
Troubleshoot/induce error	Confronts user with common error so they can learn workaround
Give assignment; see an expert solution	Allows novice to compare their solution with expert solution

Rule Training	Help user learn a rule-of-thumb that can be used to predict AI
Rule Untraining	Disabuse learner of reasonable but incorrect rule for predicting AI
Counterfactual contrast	Show how a change in input/settings leads to a change in AI behavior
Semifactual-Counterfactual sequence	Identify a pathway where making a small change does not impact outcome but a larger change does impact outcome
Mental Model Matrix	Mental model matrix
Cheatsheet	Permanent reference for later use after tutorial completed
Shadowbox	Trainee makes decisions and provides rationale within defined scenarios, and receives expert feedback on those decisions.

Module: Walkthrough

Walkthroughs are documents or videos often developed by video game players that give a minimal sequence required to complete a game, and have formed an important component of third-party generated training for video games (deWinter, 2016). Similarly, basic how-to-use-it tutorials exist for many common algorithms. The key aspect of a walkthrough is to show positive examples of usage, and the tool operating as expected. This may be an important initial component of the cognitive tutorial because many of the later modules show how things might go wrong. These error cases are critical for learning the boundaries of a tool, but may either be difficult to understand if the user does not first understand how the tool *should* be used, and it may also be demotivating or give the user the impression that the tool is error-prone. The basic steps of a walkthrough can be seen in the flowchart in Figure 2. A ‘walkthrough’ is a guide that provides a sequence of operations that allow the task to be accomplished. It also gives generic advice on how to use the tool properly for a new problem. The rationale is that after watching step-by-step instruction, many users are lost when it comes to repeating steps on a new problem.

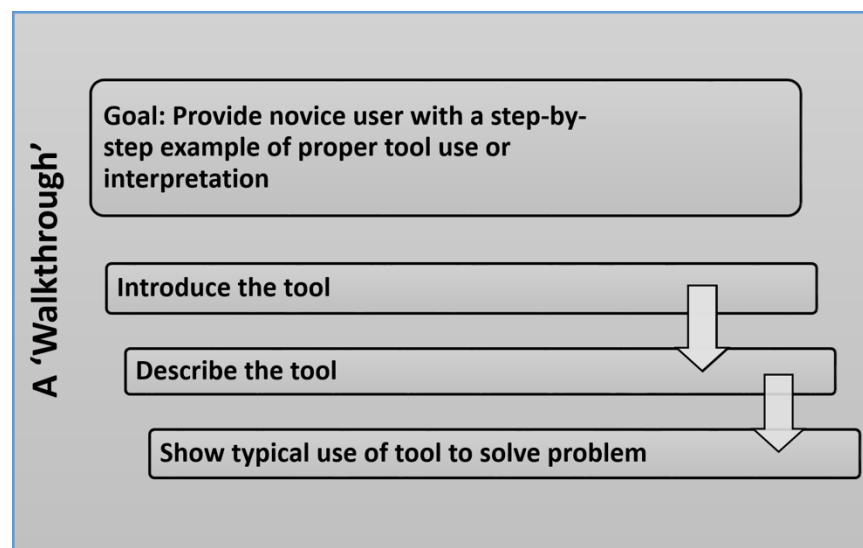


Figure 2: The flowchart of a Walkthrough learning module.

Module: Forced choice scenarios

As for forced-choice scenarios, it presents a scenario and a comparison of two alternative solutions, which may include the tasks, situations, data sets, and contexts for which the tool is well-suited, in comparison to those it is ill-suited for. It shows positive and negative usage quickly with low overhead. Figure 3 shows the goal and steps of forced-choice scenario learning module.

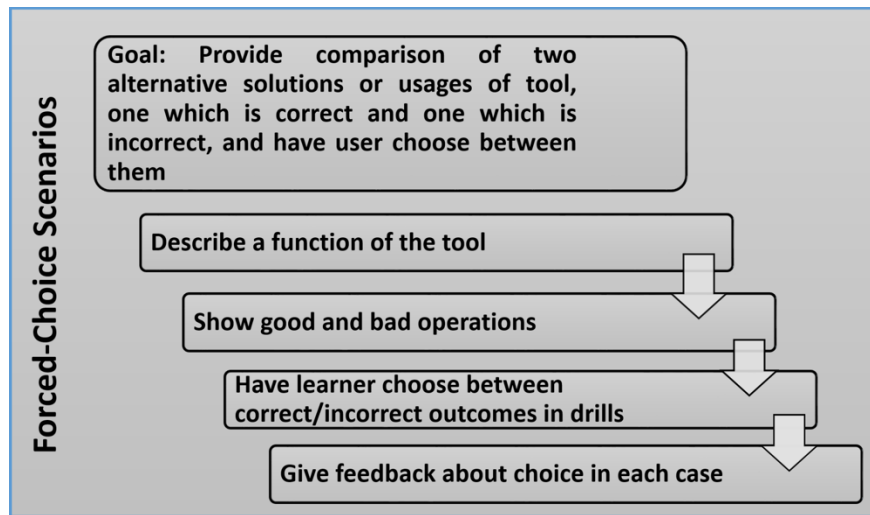


Figure 3: The flowchart of Forced-choice scenario learning module.

Module: Troubleshoot/Induce error

This module is intended to expose a user to failure modes in order for them to understand the proper and improper use of a system. For example, prior research has implemented troubleshooting to identify the best components to fix in the image captioning system (Nushi et al., 2017). Typically, errors are experienced frequently by novices without a clear correct outcome, and infrequently by experts, who may be able to identify a correction or workaround. Both of these can be sources of learning, but they can also be frustrating, especially when they cannot be easily resolved. The purpose of this module is to present a scenario with a specific problem, error, or misinterpretation, and allow the learner to discover the problem. Errors are used to highlight boundary conditions and problem areas. The advantage of this over naturally-occurring errors is that (1) a true solution can be determined and incorporated into a lesson; and (2) one does not have to wait for errors to occur.

Research suggests that errors and mistakes can provide important learning experiences (Metcalfe, 2017). For any AI tool, there are a number of common mistakes a novice can make in creating, training, interpreting, or using a model. This module type involves developing a scenario in which the user confronts a problem with the AI system and must troubleshoot the problem. If possible, the scenario may lead the learner down a garden path by encouraging them to misinterpret some element of the model, but it can also be presented as a mistake another novice user made that the learner must troubleshoot. The user should be required to attempt to fix the problem or explain how they would fix it, and following this they are given feedback

about what the problem really was. Although errors can be valuable to help identify boundary conditions, this must be balanced with examples of proper use, so that training does not become demotivating or focus too much on failure modes of a tool.

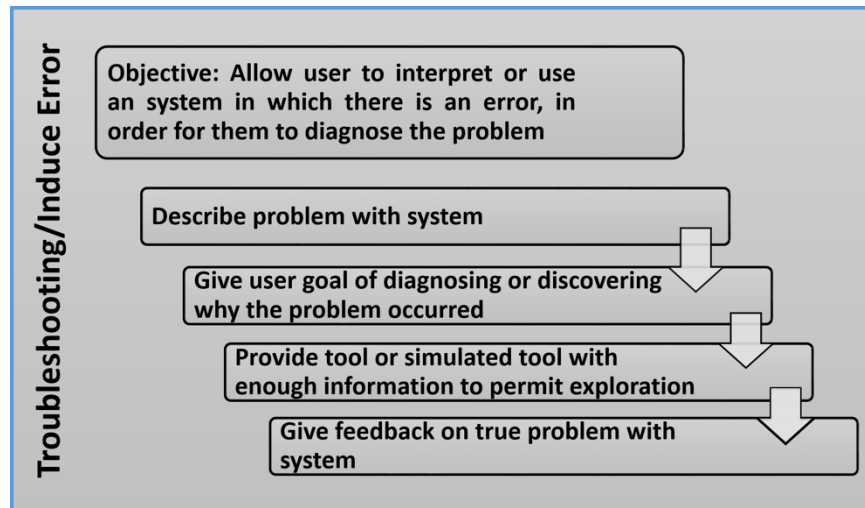


Figure 4: Troubleshooting/induce error scenario

Module: Work novel problem and see expert solution or interpretation

Expert-worked examples can be employed in a number of ways. Along with the Shadowbox method (described later), expert solutions or explanations can be employed early (fulfilling a similar role to the walkthrough) or to highlight a limitation/mistake (by encouraging the user to make a mistake and then showing the expert solution). The basic approach is for the learner to build, use, or interpret a model or tool, and compare this to an expert solution or interpretation.

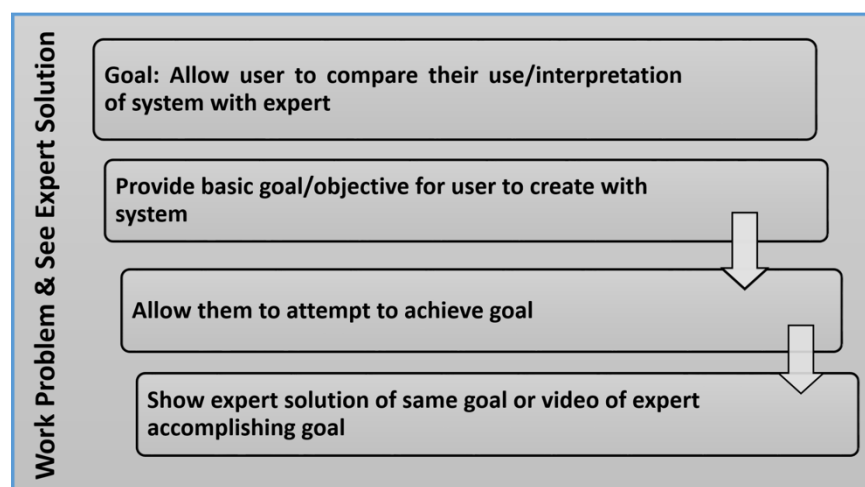


Figure 5: Flowchart of the 'work problem and see expert solution' learning module.

Module: Rule Training

Sometimes the goal of the tutorial is to teach users a rule that they can use to understand and predict the system. By ‘rule’, we mean any verbalizable statement that can be used to help predict system performance—it does not have to be 100% predictive, but rather a useful way to shortcut the sensemaking process about the system. For an image classifier, a rule might be a statement about how different features lead to different errors. For a voice-to-text classifier, a rule might be related to how different properties (accent, noise, age of speaker) lead to different errors, or it might be something like “the output is almost always spelled correctly and grammatical even when it is not correctly recognized”. For a music recommender system, it might relate to how recommendations focus on the genre of a band more than the style of the song. In all these cases, the rule might be discovered through interviews or analysis/testing of the system. On their own, individual rules may be easy to remember, but when multiple need to be applied, more deliberate training might be necessary.

The following tutorial method (Rule Training) is intended to be a systematic way to teach such rules. This method assumes the rules have been identified either through expert analysis or previous usage of the AI system. Figure 6 shows the basic flow of rule induction training from a basic explanation of the rule, to providing visual examples, to learning sensitivity of the rule, to practice.

Our basic approach to rule training involves identifying several important components: 1. A clear statement of a rule (formulated as something like an if-then statement) that describes the AI behavior, at least probabilistically; 2. Identification of visual cases (e.g., training images) showing the rule in operation; 3. An analysis of the sensitivity of the rule—including the accuracy with which applying the rule will help predict the AI performance; and 4. A restatement in words of the sensitivity of the rule. Because there may be numerous rules for any complex system, we suggest organizing this information on a ‘rule card’ that can be used for later reference (see Figure 7 for an example).

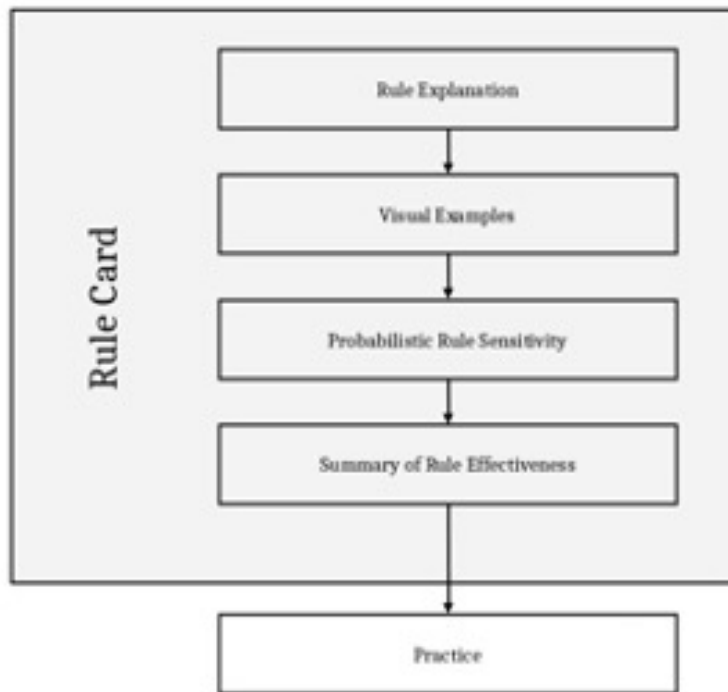


Figure 6. Basic flow of rule training.

The rule card may be used as a reference during training or at any time during the AI system's use. It may also be used by novice users of the system to refer to as needed. More examples and details are available in Appendix B (Rule Training). The rule card incorporates four main pieces of information:

Verbal Rule Explanation (A): This terse but complete verbal explanation should describe the rule, and the AI system's interpretation of the rule, both in successes and failures.

Visual Examples (B): This can be a visual representation of how the AI system succeeds and fails. It is helpful to have three examples of successes and three examples of failures.

Depiction of Probabilistic Sensitivity of the Rule (C): This representation should contain some form of a 10x10 grid, with color-coded/text-labeled components that depict system accuracy.

Summary of Rule Effectiveness (D): This is a summary restating the rule and how discriminative it is in both success and failure cases. This section can also help a user understand the context in which the rule is effective. For example, in Figure 7, the flag, when present, strongly indicates the sample will be classified as a 5. But when it is absent, the system is almost equally likely to call the image a 1 or a 5, so other rules might be needed to make a good prediction.

Following the learning process using the rule card, the next step in this approach involves practice with examples and feedback. The amount of practice is determined by the complexity of the rule and the system.

A Sometimes a 1 can be a straight vertical line, or it can have what we call a flag at the top. You can see examples below.

It turns out that the flag confuses the AI so that it often mistakes a 1 drawn with a flag for a 5.

B

Example 1s without a flag

Usually classified as 1

Example 1s with a flag

Usually classified as 5

C

Out of 100 cases of 1s without flags, 60 were classified as 1s, and 40 were classified as 5s

Out of 100 cases of 1s with flags, 97 were classified as 5s, and 3 were classified as 1s

D This rule is really good at distinguishing 5s when there is a flag. However, it's not an extremely strong indicator of when the AI system will classify the digit as a 1 when there isn't a flag. Therefore, this rule is discriminative for 1s classified as 5s.

A 1 without a flag might be drawn with a straight up-and-down vertical line, or it might be drawn slanted to the left or the right.

It turns out that the flag confuses the AI so that it often mistakes a 1 drawn with a slant to the left for a 5.

Example 1s vertically straight or slanted to the right **Examples 1s slanted to the left**

Usually classified as 1

Usually classified as 5

Out of 100 cases of 1s vertically straight or slanted to the right, 97 were classified as 1s, and 3 were classified as 5s

Out of 100 cases of 1s slanted to the left, 81 were classified as 5s, and 19 were classified as 1s

This rule is really good at discriminating between 1s classified as 1s, and 1s classified as 5s based on the angle of 1s without flags at the top.

Figure 7 Example Rule Cards for rules identified in an MNIST classifier.

Module: Rule Untraining

When interacting with an AI system, users will often engage in *cognitive anthropomorphism*: treating the AI as if it performs cognitively like humans do. This misconception may lead users astray. Potential misconceptions would have been identified during data collection in the tutorial development process. In these cases, a tutorial may help a user to unlearn, and provide reasoning or rationale for why an intuitive rule should not be used, or even that it works in the opposite

direction than pre-supposed. For example, a user might think the GPS routing system prefers interstate highways over backroads, even if it does not, and may be misled because most routes are faster and shorter via highways. Simply telling them that this ‘rule’ is not built into the system may not be enough to help them understand why it is not true. This training can use many of the same approaches as rule training, but it may be more difficult to provide a compelling explanation for why a rule is not true.

We have identified at least two special cases that might benefit from rule untraining. The first is reliance on a rule does not work. Here, an alternate causal explanation might be needed to explain why the rule appears to work, but is not a good account. In the GPS case, showing that the algorithm prefers shorter/faster routes might be enough to convince the user that there is no built-in preference for interstate highways, and that the choice comes because of estimated time or distance. The second case is one in which the condition of the rule actually is predictive, but in the opposite direction from what the user is likely to suppose. For example, a GPS user might think that an algorithm’s route optimizes distance over time, when the opposite might be true. This kind of learning module might look more like the rule training module (because they are essentially learning a rule), but might take special care to acknowledge the potential misunderstanding and explain why it is incorrect.

The following tutorial method (Rule Untraining) is intended to be a systematic way to point out such a non-rule. This method assumes the non-rules have been identified either through expert analysis or previous usage of the AI system. Similar to the Rule Induction Training process, Figure 8 shows the basic flow of the rule untraining tutorial from a basic explanation of the non-rule to providing visual examples to learning the lack of sensitivity of the rule to practice.

To untrain users about an ineffective rule, a tutorial must acknowledge and describe the ineffective rule, but may also need to provide an alternative account for why the rule may seem to be effective, or demonstrate via a visualization of probabilistic rule sensitivity that it is not effective. After this, the training can proceed much like the rule training method, but with a special focus on the ineffectiveness of a hypothetical rule and the potency of the alternative explanation. This comparison may be used throughout training, especially when showing examples: examples can be selected to show how the rule is ineffective once the alternative rule is controlled for.

The opposite-effect rule untraining proceeds in a similar way, although a secondary alternative explanation may not be needed, because it is simply the opposite of the initial rule. The incorrect rule should first be acknowledged and described, along with the proper interpretation, to help understand the proper rule. Following this, examples illustrating the rule, rule sensitivity, and training can proceed much like in the rule training module. For example, in a GPS example, a user may mistakenly believe that the GPS estimates what the traffic congestion will be like in the future to plan long routes, even if it does not. In this case, teaching that the router does not use traffic information trains the rule and untrains the misconception. Example flowcharts of these modules are depicted in Figure 8. Figure 9 shows an example rule card. More examples and details are available in Appendix B (Rule Untraining).

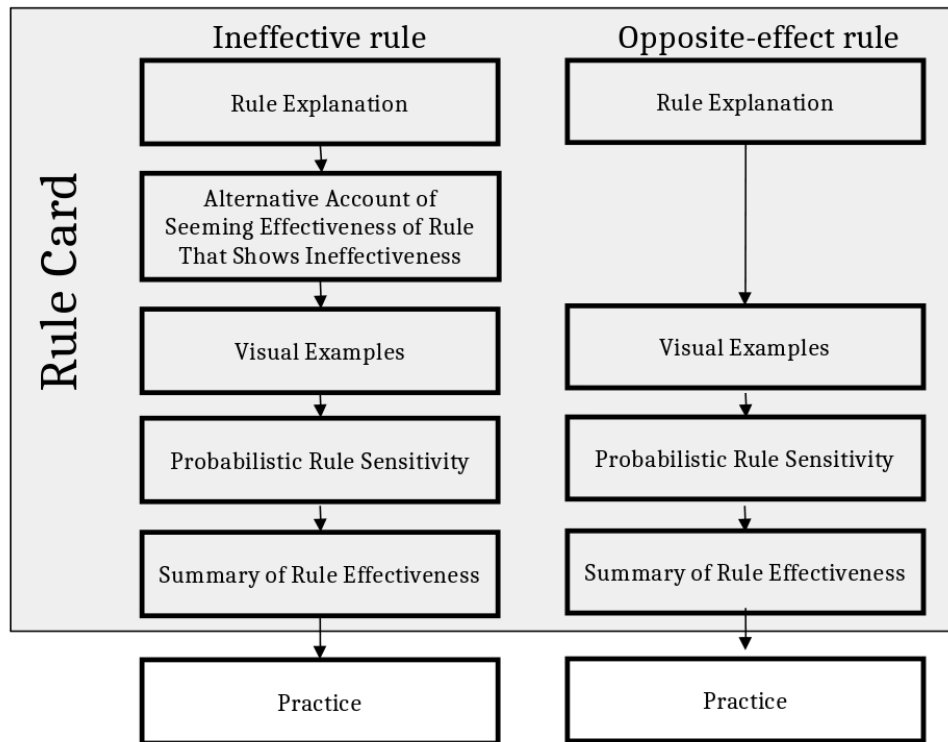


Figure 8 Basic flow of rule untraining.

A 3 might be drawn with a pigtail on the bottom, or it might be drawn without a pigtail. You can see examples below.

You might imagine that this would confuse the AI so that it might often mistake a 3 with a pigtail for a 2, but it doesn't.

Example 3s without a pigtail

Usually classified as 3, but not by much more than chance

Examples 3s with a pigtail

Usually classified as 2, but not by much more than chance

Out of 100 cases of 3s without pigtails, 53 were classified as 3s, and 47 were classified as 2s

Out of 100 cases of 3s with pigtails, 48 were classified as 3s, and 52 were classified as 2s

Contrary to what we might think, this rule doesn't do a good job distinguishing 3s with a pigtail with 2s.

Figure 9. Example Rule untraining Cards for an ineffective rule.

Module: Counterfactuals and Contrasts

Contrasts and counterfactuals have been identified as a critical method for explaining AI decisions (Miller, 2019; Mueller et al., 2021). In this context, a counterfactual is an exploration of an alternative situation in which a given fact (e.g., some visible feature of an image) differs, normally with different consequences. For a counterfactual or contrastive explanation to be

effective, it must support relatively straightforward causal reasoning: a counterfactual case must consider a situation where a small number (maybe a single) feature differs, and this difference is powerful enough to cause the system to behave differently. Complex counterfactuals make tracing the cause of change much more difficult. For example, if one knows that an image of a red bird with a crown and a short black beak is labelled as a cardinal, a counterfactual that changes only one feature (color, crown, or beak) is more informative than one changing all three features.

Sometimes the goal of the tutorial is to teach users a rule that is critical for understanding the system, such that if a condition of the rule changes, the system produces a different answer. The following tutorial method (Counterfactual Contrast) is intended to be a systematic way to teach such a rule. This method assumes the rules have been identified either through expert analysis, or systematic algorithms that inform developers about the rule's importance. As with earlier 'rule training' modules, the rule does not need to be 100% discriminatory, but should at least provide a reasonable improvement in accuracy if used. Figure 10 shows the basic flow of counterfactual contrast training from a basic explanation of the rule, to providing visual examples, to learning sensitivity of the rule, to practice.

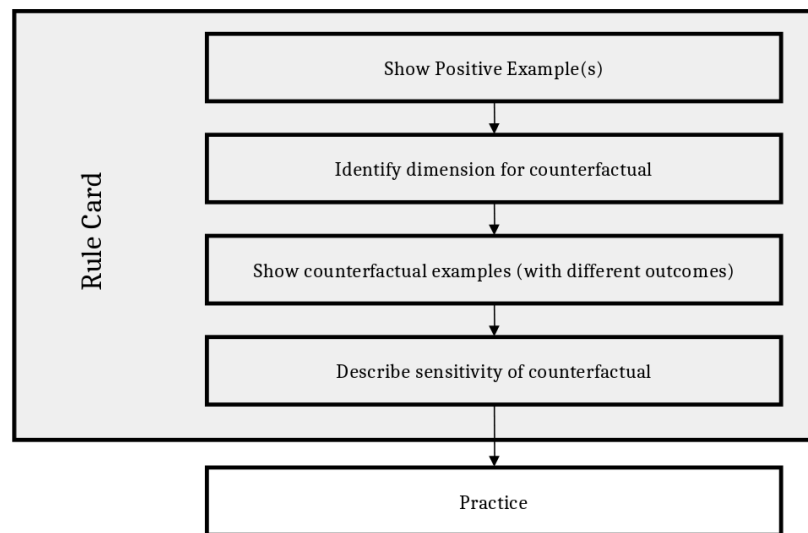


Figure 10. Basic flow of counterfactual contrast training.

Like the rule training methods, a counterfactual comparison can be organized in terms of a single-page "rule card" that incorporates the first four stages of the counterfactual contrast process: a verbal Rule Explanation (A), Visual Examples (B), a Depiction of Probabilities (C), and a summation of the Sensitivity of the rule (D). The rule card may be used as a reference during training or at any time during the AI system's use. It may also be used by novice users of the system to refer to as needed. Figure 11 shows an example rule card. More examples and details are available in Appendix B.

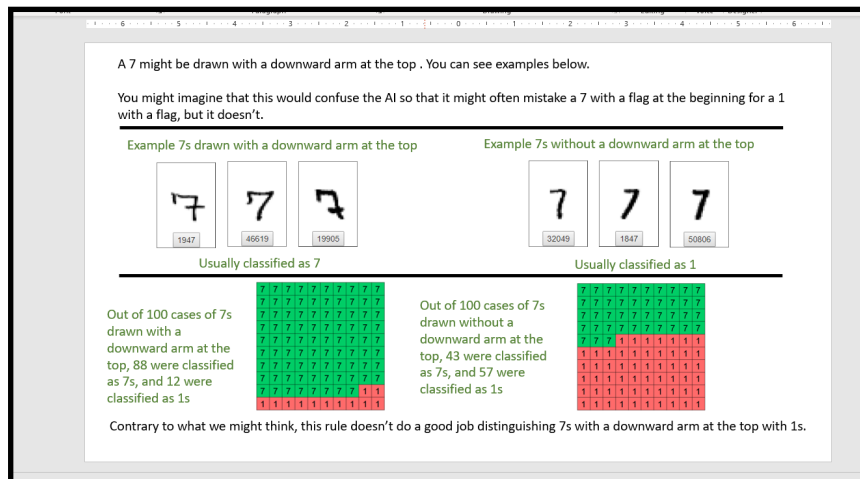


Figure 11. Counterfactual Example Rule Card.

Module: Semifactual-Counterfactual Sequence

Kenny and Keane (2020) investigated the use of both counterfactuals and semi-factuals to help explain AI systems. In their framework, a semifactual refers to a contrasting example that does not change the outcome, whereas a counterfactual is one in which the contrasting example does change the outcome. For example, if an AI has learned to classify furniture, a seat that is 24 inches wide would be classified as a chair, while a similar seat 55 inches wide might be classified as a love seat (the counterfactual outcome of changing the width feature). But a seat that is 30 inches wide might still be classified as a chair (semi-factual). By showing different examples along the spectrum of width, the user can develop a more precise understanding of the sensitivity of a particular feature, and the particular boundary conditions between the two classes.

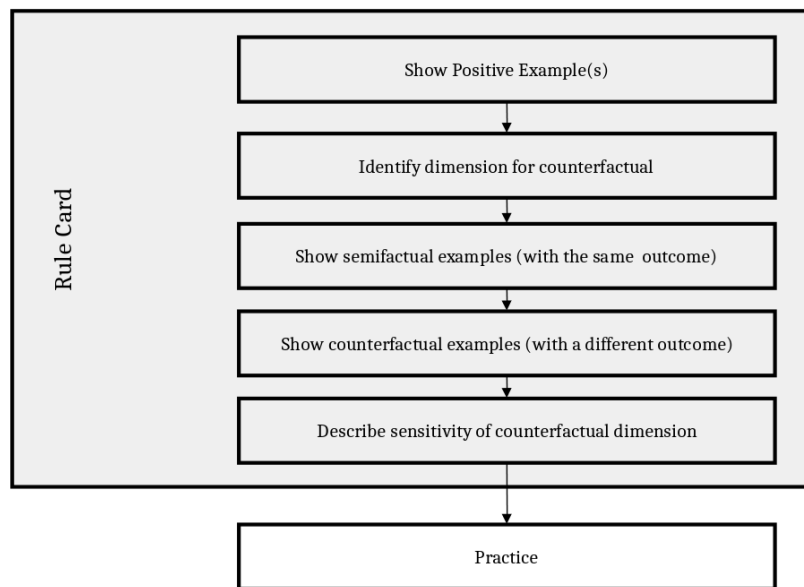


Figure 12. Flowchart depicting a training sequence for a semifactual-counterfactual sequence.

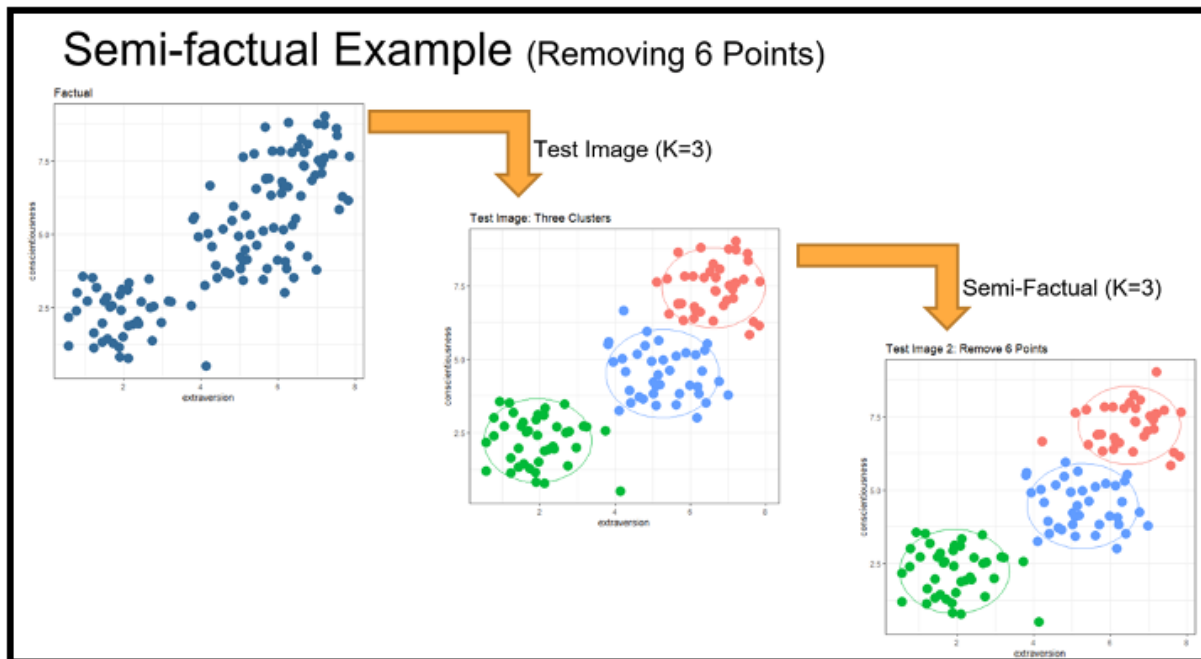


Figure 13. Example training for semifactual-counterfactual sequence. Here, points are removed to first show how the same basic clustering pattern is achieved. Subsequently, as more points are changed, the outcome eventually changes, which helps highlight the sensitivity to change of the algorithm

Module: Mental Model Matrix

A mental model is a concept, framework, or worldview that tries to explain a thought process and help individuals function in the world. The definition of the Mental Model Matrix we discuss here was developed by Borders, Klein & Besuijen (2019), including the following four components (also see the flowchart in Figure 14):

- How a system works (e.g., parts, connections, causal relationships, process control logic)
- How a system fails (breakdowns and limitations), which is important for identifying steps for refinement, knowing system reliability in multiple settings, and for increasing user confidence toward a system (Nushi et al., 2018)
- How to make a system work (e.g., detecting anomalies appreciating the system's responsiveness, performing workarounds and adaptations)
- How users get confused (the kinds of errors people are likely to make).

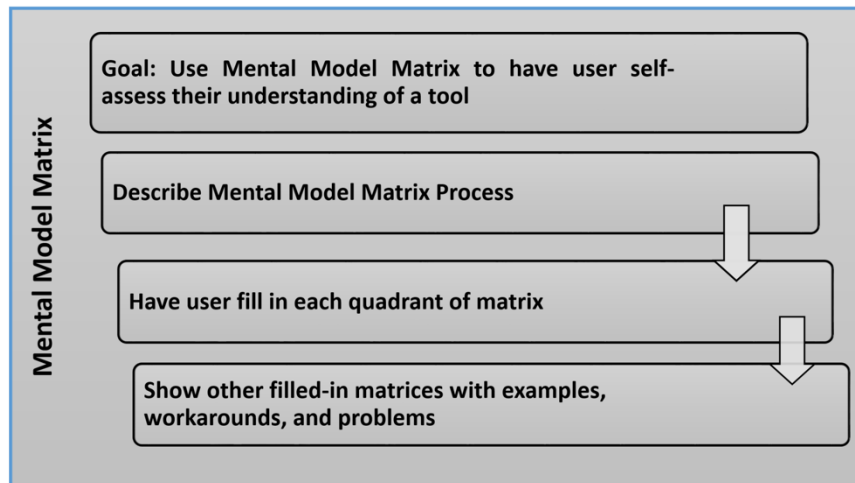


Figure 14. The flowchart of Mental Model Matrix tutorial approach.

Using the Mental model matrix as a tutorial may be effective as it does not simply teach facts, but attempts to change the way a user understands the intricate workings of a tool or system. The rationale of using a mental model matrix as a learning module is that it can help the user’s self-explanation and sensemaking holistically through four components including positive and negative sides of a system and of a personal self-reflection. This model helps a user to think comprehensively and simultaneously while seeing and answering the four-quadrant questions (Figure 15).

Figure 15: The template of a Mental Model Matrix.

	Positive (+)	Negative (-)
System	How the system works: Parts, connections, logics, causal relationships	How the system fails: Breakdowns and limitations
Person	How to make the system work: Detecting anomalies, appreciating the system’s responsiveness, performing workarounds and adaptations	How users get confused: The kinds of errors people are likely to make

In general, these modules include examples of how the system can be used successfully and illustrations of how it can be misused, in order to help the user understand the boundary conditions of the system.

Module: Cheatsheet

Mueller et al. (2021) argued that explanations are rarely one-off: explanation is a continuing process that develops over time and leads to better understanding as the user learns. Many of the modules here are useful as introductory materials, and so do not easily support repeated use for training. We suggest that for any module, a cheatsheet can be developed that can serve as an easy

resource for users later. Cheatsheets have been examined in educational settings, and they have shown to be detrimental for later test-taking (Dorsel & Cundiff, 1979); yet they have been advocated as important resources for organizational knowledge (Halverson & Ackerman, 2008). They have also proven to be some of the most widely used resources for learning different programming paradigms (see <https://www.rstudio.com/resources/cheatsheets/>). In the Appendix, we have included an example cheatsheet for clustering algorithms that helps identify all the basic rules, strengths, and weaknesses, on a single reference page.

Module: Shadowbox

Klein and colleagues (Klein, Hintze, and Saab, 2013; Klein & Borders, 2016) have advocated and evaluated the Shadowbox method for developing cognitive skill and helping novices develop expertise in a system. Although this approach is general and not targeted to intelligent software tools, it can still be useful when adapted to Cognitive Tutorials for AI. Other resources provide detailed instructions for applying the method, and the Shadowbox approach is similar to a number of the methods we have already described that either have users make choices and see outcomes or examine expert solutions to problems. However, the approach is a bit more systematic. Once a learning objective is identified, it involves:

1. Identifying decision points that highlight difficult choices or decisions relevant to the AI, via incident-based Cognitive Task analysis interviews.
2. Identifying candidate alternatives that might be chosen at this point.
3. Collecting expert accounts of their decisions, with feedback on why each of the alternatives is good or bad
4. Developing a scenario in which the learner can make the choice and give their explanation of why it was good
5. Showing feedback based on the chosen (and other best-case non-selected actions) given by a panel of experts.

Summary

In summary, this section demonstrates 10 example training modules that could be adapted for many cognitive tutorials. Other approaches are certainly possible, or hybrids between these as well. Implementation of these can sometimes be very low-tech (screenshots on websites or printed documents), or could be a mixture of text, video, web forms, and dynamic tools--perhaps implemented within the tools already being used.

4. Discussion

Altogether, the goal of developing cognitive tutorials for AI is to serve as a means for global explanation of an AI system, rooted in local explanations of specific cases, with attention to how the system works in context and how experts use and work around the limitations of the system. It should:

- Add to existing training, focusing on advanced/expert use of system, and not replace existing training (unless the existing training is deficient)

- Focus on the cognitively challenging elements of system use, not buttonology or first steps.
- Identify cognitive misconceptions, common problems, and workarounds experienced users have developed
- Use experiential learning approaches to show users **how** to work with systems, rather than providing a deep understanding of underlying analytic algorithms.

In the next section, we include some basic advice and recommendations for implementing a CT.

Recommendations for implementing a CT for users

Algorithmic XAI may not be sufficient.

One goal for explainable AI systems is that they help the user understand how the algorithm is working. The greatest rationale for implementing a CT is that the information provided by ‘Explainable’ algorithms is inadequate for users to understand how to use the system. Thus, a CT may augment rather than replace necessary interface and output algorithms that support explainability. One limitation of explainable AI systems is that they tend to focus on local explanation and justification—why certain decisions were made for certain cases. This information is not present directly in tutorials, although examples may give a user an understanding of why a decision was made without a complex algorithm. However, local justifications and global explanatory tutorials are likely to be a powerful combination for helping users understand systems.

A CT is for non-trivial intelligent software systems and system functions.

Experiential training has proven effective in many contexts, and so it is certainly true that experiential training could be effective for many ordinary functions of software. We focus on intelligent software functions because users are especially apt to need an accurate mental model of those systems, while at the same time non-experiential methods for training users in the system are likely to be highly technical and beyond what most users will tolerate. For straightforward software functions, scenario-based learning may be inefficient for the user as well, because he or she may simply want to look up a function reference in a detailed help system. In addition, limited resources mean that investments in training need to be prioritized. Thus, it is reasonable to focus experiential training on the aspects of the tool that are either most likely to be beneficial, or most likely to cause problems if there is no experiential training. So, although a CT could cover non-intelligent software functions, it may only sometimes be appropriate for them.

A CT is not an introductory coursebook.

When designing a CT, developers must gauge the experience a typical user will have. This will depend on the tool, the user population, and the existence of training manuals and courses. Often, there is little introductory-level training material, or conversely, highly detailed training is embedded in the system's software. Both can lead to misconceptions and improper use. A CT is intended to focus on developing a functional and accurate mental model of underlying intelligent algorithms, but if a user has no way of understanding the basic functions of the tool, they may never get to the point where a calibrated mental model matters.

Development of a CT does not replace usability testing.

Usability concerns are almost always a huge challenge for special-purpose software with relatively small user bases, which we anticipate the CT being most valuable for. The process of developing a CT can unearth many usability issues, but there is a distinction between the kinds of usability issues revealed by standard usability testing, and the types of problems the CT should focus on. Usability testing is often predicated on the notion that the interface can be made easier to use. The CT is intended to support the types of functionalities that cannot be changed, that cannot be made easier, and that will necessarily require experience and training to get right. By analogy, user testing might help one design the ergonomics of a new musical instrument, but it will never make the instrument so easy to play that even a novice can perform masterfully.

Explanations are not one-off.

Explanations are not a one-off thing-- a user's understanding of a system evolves and improves over time. A CT should recognize this. One way to design a CT for this is to develop a CT curriculum that covers basic to advanced usage. Here, the goal is not to provide only a basic introduction, but also provide training that is useful once more experience is gained. This involves experiential training that can be referenced throughout use when certain issues arise. Finally, modules/lessons could be concluded with cheat-sheets or other mnemonic aids (like the 'rule cards' we developed for rule training) that could be used as long-term aids for using a system.

5. Conclusion

In this Report, we have provided concepts, methods, and examples for implementing cognitive tutorials that can meet explanatory needs of AI users. The goal of these tutorials is to augment and support users, and to supplement existing low-level training and dynamic algorithmic XAI explanations that might be a part of a system.

Appendix A Sample Cognitive Tutorial Authoring Example

In this Appendix, we demonstrate the different steps we engaged in to complete a cognitive tutorial for a simple k-means clustering algorithm. This went through steps from identifying learning objectives to developing and evaluating the tutorial.

Step 1: Identify Learning Objectives

The primary goal for Step 1 (Identify Learning Objectives) is to support the development of cognitive tutorial via identifying vignettes, stories, and examples that can be used or adapted to form the basis for cognitive tutorial lessons. The method to accomplish it is to examine an ensemble of sources. First of all, we identify more than 30 sources, including how-to websites, on-line communities, textbooks, and videos. We then organize the sources based on possible problems and/or challenges users may have. For each of the problems, we identified possible solutions for each of the problems in Table A1.

Table A1. Examination of web-based forums.

Problem/Challenge	On-line Source Format	Possible Solution
What type of system/tool is used? What is the targeted system?	R website	R Shiny
How to use the system/tool? How to initiate the clustering?	Observation of "First 20 Minutes"	
How the system represents information?		
Identify the data sources	Data analysis	personality online pre-screening questionnaires
How to tell if data is "clustered" enough for clustering algorithm?	stats.stackexchange.com forum	use the Gap statistics. Basically, the idea is to compute a goodness
Normalization of network data (clustering algorithms)	stats.stackexchange.com forum	designed to work on continuous variables, no sense for IP address
K-means clustering scaling	stats.stackexchange.com forum	
Clustering a dense dataset	stats.stackexchange.com forum	select the method and the final clustering solution by which possible
How would PCA help with a k-means clustering analysis? (When should I use PCA?)	stats.stackexchange.com forum	1. Doing PCA before clustering analysis is also useful for dimensionality
Data Preparation for Cluster Analysis	stats.stackexchange.com forum	data normalization and removing correlation among data are often
How the tool work, and how it actually does work?	Algorithm analysis	R Shiny source code
When would I use EM instead of k-means?	stats.stackexchange.com forum	there will be uncertainty about your cluster assignments so it is difficult
The problem of how to choose the number of clusters	Book (The Element of Statistical Learning)	Kmeans clustering is a top-down procedure, while other clustering algorithms
Partitioning clustering	Book p.229 (Encyclopedia of Machine Learning)	K-means is the most widely used clustering algorithm. It constructs
Are there cases where there is no optimal k in k-means?	stats.stackexchange.com forum	there is no cluster structure in the data. However, clustering in very
K-means clustering has shortcomings in breast cancer clustering	Book p.514 (The Element of Statistical Learning)	1. it does not give a linear ordering of objects within a cluster; 2. as
K-means will not work well when the clusters are non-convex, spherical	Book p.544 (The Element of Statistical Learning)	K-means use a spherical or elliptical metric to group data points; spherical
When clusters are stretched-out banks, the k-means algorithm fails	Book p.21 (Mining for Strategic Competitive Intelligence)	density-based clustering approaches (see subsequent paragraphs)
The limitation of a traditional K-means algorithm (unable to cluster overlapping clusters)	Book p.63 (Network Intrusion Detection using Machine Learning)	It is observed that the advantage of SAE (Stacked Auto-Encoder)
the steps for K-means clustering	Book p.85 (Natural Computing for Unsupervised Learning)	1. Select K points as initial centroids.
The huge advantage of k-monoids over k-means	Book p.22 (Mining for Strategic Competitive Intelligence)	the first is less susceptible to outliers than the latter, as the cluster
The huge advantage of density-based clustering schemes (e.g., DBSCAN)	Book p.25 (Mining for Strategic Competitive Intelligence)	clusters do not necessarily have to be cloud-like in order to be discovered
Less desirable properties of K-Means: (a) the initial setting, (b) the	Book p.224 (Core Concepts in Data Analysis)	(a) the initial setting, i.e. the number of clusters K and initial positions
Can the centroids be incorrect even if there is convergence?	CodeCademy Q&A	Yes, absolutely. K-Means clustering finishes once the centroids no
How to visualize K-means clusters in 3D?	YouTube Video	Visualizing K-means algorithm in 3D
Real-time simulation of the K-means clustering algorithm	YouTube Video	K-Means Clustering Example, using different values for n and k
How to produce a pretty plot of the results of k-means clustering?	stats.stackexchange.com forum	library(cluster); library(fpc)
Insufficient interpretation aids	Book p.225 (Core Concepts in Data Analysis)	

One important source of data collection can be the team of developers who produce the system/tool. In this k-mean clustering example, one valuable source includes researchers who are familiar with the algorithms and test the system to understand and improve it. For this reason, we conducted an interview with the principal developer and identified four possible problems and/or misconceptions users may have in Table A2.

Table A2. Interviews with the developer/researcher

	Problems/Misconceptions
1	Long skinny data
2	Different variances
3	Handily categorized & binary variables
4	Euclidean vs. Manhattan distance

Step 2: Develop Models of the Reasoning of Expert and Non-expert Tool Users

The goals of Step 2 include evaluating the usefulness as guidance for future developers, evaluating the cognitive mismatch between the user's mental model and the tool's operation, and avoiding misapplying the rules of another similar tool to the current tool. We found it valuable to observe users exploring a learning tool for the first time. Such observations can help identify some of the stumbling blocks someone experienced when they faced the tool for the first time.

To do that, we asked undergraduate students in a Research Method class who had no prior knowledge on k-means clustering algorithms to experience the learning tool we developed. We observed their "First 20 Minutes" of experiencing the learning tool in class. Then, we conducted a group interview with them and identified nine problems/concerns in Table A3.

Table A3. Observations of users' "first 20 minutes."

In-class focus group data collection	
1	Needs to be a certain type of data (can't use distance, need to use features)
2	Sometimes it gives you strange output, if you run it more than once
3	It can come up with different clusters for the same data. A hidden cluster available for the 4th cluster would appear as the 3rd cluster
4	Specifying number of dimensions/distinguishing between dimensions and K
5	Fit measures are 'convoluted'; betweenSS/totalSS
6	Distance measures—could be challenging
7	No standardized form depending on input
8	How do you determine the best fit, especially versus K. Determining a good fit is confusing

In addition, conducting interviews with experienced users can provide us with guidance regarding the cues they look at in the tool and alternative ways in which the tool could have or should have been used. To do this, we conducted interviews with four experienced users who were graduate students and had taken Applied Statistics Analysis for Psychology course. Then, we organized the interview results and identified 27 questions and problems users had. We

categorized them based on the four functions of a cognitive tutorial for understanding AI systems. Finally we identified possible learning objectives in Table A4.

Table A4. Identified problems and possible learning objectives experienced users had via interviews.

Function	Problems	Possible Learning Objectives	Interview
Representation and Modeling	Not familiar with the learning tool	learn how to use the learning tool (e.g., Kmean in R); individual	P1,2,3,4
	how to interpret/compare pairs in the ggpairs		P2
	preferences of reading texts (including tables	understand users' preference/tendency before developing pers	P2,4
	Some people need more time to comprehend	(get deep learning) while others prefer to skim through it	P4
Data Handling and Data Generation	Not familiar with the data and how they were	Source of data description	P4
	Need to understand the subset data in order to	learning of parameters (variables)	P1,2,3,4
	One variable drives the whole clustering	understand parameters and data range	P2
	Before performing clustering, they had to make sure the data/variables were comparable		P1,2,3,4
	Fake data is easier to interpret		P1
	Experimental data is easier to interpret/understand		P2,3,4
Understanding Computation and Algorithms	The clustering is affected by scales (ranges of	transformation: scaling & normalization	P1,2,3,4
	Not clear about the K-means algorithms	Providing the opportunity of algorithm learning	P1,2,3
	How to select parameters?	select parameters	P1,2,3,4
	How to determine # of clusters?	choose # of clusters	P1,2,3,4
	Chose 2-3 groups only because it is easy to interpret	Goodness of fit (learn how many variants are covered)	P1,4
	Can interpret the plots, but not sure whether it	Not only how to interpret the results is important, but how to fit	P1
	need to learn a more efficient way of clustering	appendix of other clustering methods	P2
	seeing the trees and missing the forest (just looking	need specific/immediate guidance/instruction	P2,3
	just trial and error for parameter selection	need to learn pre-clustering method for parameter selection (e.g.	P3
What is K-means doing?		P3,4	
Need more time to obtain comprehension or deep learning (others might just skim through it)		P4	
Output, Display, and Visualization	how to interpret ggpairs results (not just over	Learn to interpret resulting plots	P1,2
	need to look into X & Y axes	Learn to interpret resulting plots (including x & y axes)	P1,2,3,4
	Reading vs. Visualizing oriented individuals	Adding figure captions	P2,4
	dig into matplotlib/plot, other than just ggpairs	provide plots of other formats	P3
	Prefer reading texts/numbers than figures		P2,4
	Whether the tool can be more useful than just presenting the clustering results		P1

Step 3. Identify Gaps in the Training

The goals of Step 3 Identify Gaps in the Training are to craft user problems that highlight issues, misconceptions and problems as they arise, and to facilitate the creation of learning lessons. To do that, we organized the documents reviewed (Table A1 & A2 in Step 1) and the problems/challenges expert and non-expert users encountered in the interviews (Table A3 & A4 in the Step 2) to reveal the possible gaps in the training. Table A5 provides the integrated results of the learning problems users had, possible learning objectives for users and the sources to integrate the overall results based on system functions.

Table A5. Integrated results for identifying user problems and learning objectives.

Function	Problems	Possible Learning Objectives	Source
Representation and Modeling	Not familiar with the learning tool; What type of system/tool is used? What is the targeted system?(O1) How to use the system/tool? How to initiate the clustering?(O2) How the tool work, and how it actually does work?(O11)	What type of system to use and how to use the learning tool (e.g., kmeans, ggpairs in R); individual training guide	O1,11 ,P1,2,3,4
	How the system represents information?(O3) preferences of reading texts (including tables) vs. visualizing plots	understand users' preference/tendency b	O3,P2 ,4
	Some people need more time to comprehend (get deep learning) while others prefer to skim through it		P4
Data Handling and Data Generation	Not familiar with the data and how they were generated; Identify the data sources(O4); How to tell if data is "clustered" enough for clustering algorithms to produce meaningful results? (O5)	understand data & provide data description with/without algorithm interpretation (story telling)	O4,5,C 1, P4
	Need to understand the subset data in order to interpret plot results	learning of parameters (variables)	P1,2,3,4
	One variables drives the whole clustering	understand parameters and data range	P2
Understanding Computation and Algorithms	Not clear about the algorithms; sometimes it gives you strange output, if you run it more than once (C3); Fit measures are 'convoluted'; betweenSS/totalSS(C6); Distance measures—could be challenging(C7); K-means will not work well when the clusters are non-convex, such as concentric circles (O17) When clusters are stretched-out banks, the k-means algorithm will inevitably fail to produce good results (O18) The limitation of a traditional K-means algorithm (unable to cluster complex and high-dimensional data) (O19) Less desirable properties of K-Means: (a) the initial setting, i.e. the number of clusters K and initial positioning of centroids, (b) instability of clustering results with respect to the initial setting and data standardization, and (c) insufficient interpretation aids. (O23); Sometimes it gives you strange output, if you run it more than once (C3)	Algorithm learning (steps, advantages, shortcomings of K-means)	O17,18 , 19,20,23,C3,6, 7,P1,2,3 C5,P1, 2,3,4
	How to select parameters? Specifying number of dimensions/distinguishi	Parameter selection	
	How to determine # of clusters? It can come up with different clusters for the same data. A hidden cluster available for 4th cluster would appear as the 3rd cluster. Chose 2-3 groups only because it is easy to interpret via graphs (P1,4) Are there cases where there is no optimal k in k-means? (O15) Can the centroids be incorrect even if there is convergence? How? (O24) Clustering a dense dataset (O8)	choose # of clusters; partition clustering	O8, 13, 14,24, C4,P1, 2,3,4
	How do you determine the best fit, especially versus K. Determining a good fit is confusing.	Determine the best fit	C9
	Can interpret the plots, but not sure whether it was correctly clustered	Not only how to interpret the results is im	P1
	need to learn a more efficient way of clustering people; When would I use EM instead of k-means?(O12) K-means clustering has shortcomings in breast cancer clustering application (O16); The huge advantage of k-monoids over k-means (O21); The huge advantage of density-based clustering schemes (e.g., DBSCAN) over k-means and EM (O22)	Comparing to other clustering methods	O12,21 , 22,16, P2
Function	Problems	Possible Learning Objectives	Source
Understanding Computation and Algorithms	The clustering is affected by scales (ranges of data); No standardized form depending on input(C8). Data Preparation for Cluster Analysis (O10)	Data preparation; transformation: scaling & normalization of data	O6,7,10, C8,P1, 2,3,4
	Output, Display, and Visualization	how to Interpret ggpairs results (not just overlap); How to visualize K-means clusters in 3D? (O26) Real-time simulation of the K-means clustering algorithm (O27); need to look into X & Y axes	Interpret the visualization of k-mean clusters (including x & y axes); have them to generate plot caption
	dig into matplot/plot, other than just ggpairs; How to produce a pretty plot of the results of k-means cluster analysis? Insufficient interpretation aids (O29); Reading vs. Visualizing oriented persons	How to produce pretty plots; learn the plots from a new way; structured way to present plots Provide sufficient interpretation aids (e.g., appendix, figure captions)	O28, P3 O29, P2,4

Step 4: Map the Modules onto the Training Gaps

The goal of Step 4 is to specify a subtask or reasoning steps in order to map the modules onto the training gaps and pilot a training tool. To do that, we developed an interview protocol, which includes procedures, questions and rationales, using Cognitive Task Analysis (CTA) and Think Aloud Methods, and then we conducted the interviews in three time-points to four experienced users to demonstrate the steps for k-mean clustering algorithm training as shown in Table A6.

Table A6. Interview Protocol using Cognitive Task analysis and Think Aloud Methods with experienced users/experts.

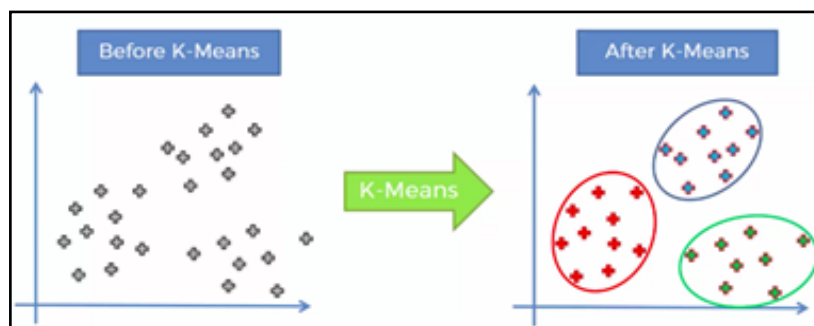
Time Period	ThinkAloud Question (for material development)	Rationale
Before the use of the learning tool	Present data (Excel) and ask participants to predict/draw the results of K-means clustering.	Prediction of the learning results
During the use of the learning tool	Show participants specific examples of the K-means clustering tool being used. Ask participants to use Think Aloud strategy: Have them comment or interpret what is being done (for the cues they look at to understand its parameters and settings, and alternative ways in which the tool could have or should have been used).	Observation of natural learning sequence; knowing the misapplying the rules of another similar tool is exactly the type of mismatch between mental model and tool operation that an EUG should focus on
Post interview	Talk about what are the specific conceptual challenges faced by you. Discuss a way you used the tool when you were first beginning that you do not do anymore. Talk about a time when you used the tool to solve a problem you wouldn't have been able to do without it. Have you observed the difference between your prediction of the results before using the tool and the results you have learned via the tool? Explain the reasons. Discuss the differences of these three data/tasks. What was a misconception you had about the tool that you no longer have? How has the tool changed the way in which you work? What are the questions/problems/challenges you still have?	Test some of the EUG concepts and get direct immediate feedback; Understanding the conceptual/mental challenges facing participants
	Understanding, trust, reliance, would work	

Appendix B Example Modules

In this Appendix, we show example training modules we have developed for many of the modules discussed in Section 3. It is important to note that these are not complete--they would normally be augmented with initial instructional videos, more detailed practice, and interactive applications, but they give basic and useful starting points. We have done this for two different systems: a k-means clustering algorithm applied to randomly-generated points, and an image classification algorithm applied to the MNIST (handwritten numerals) data set.

Module: A 'Walkthrough'

K-Means clustering algorithm: try to identify if there are clusters of points in the data. A cluster is a set of points that are similar to one another, but dissimilar to points in other clusters.



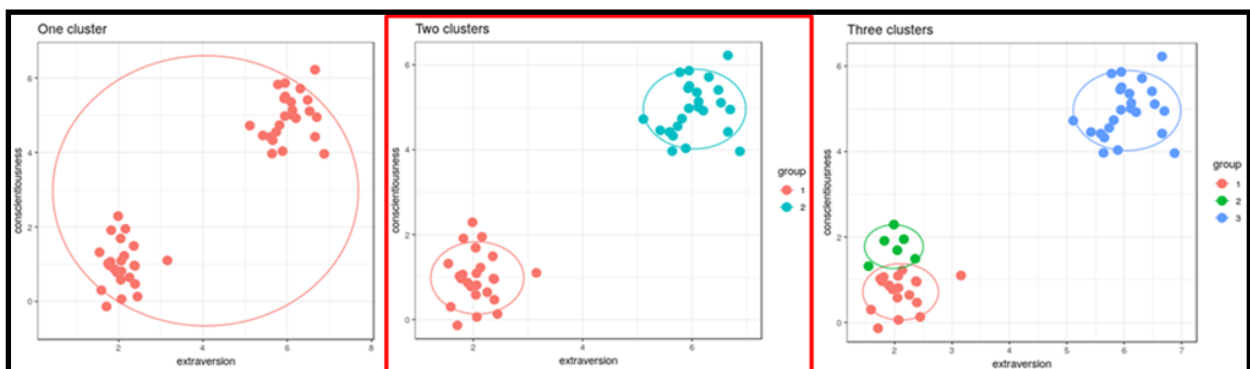
Criteria for knowing whether the solution is good:

- 1 The clusters should have little overlap (**segregation**)
- 2 There should be gaps between clusters (**separation; Between/Within SS**)
- 3 The more clusters there are, we require stronger evidence (**simplicity**)

Module: Forced-Choice Scenarios

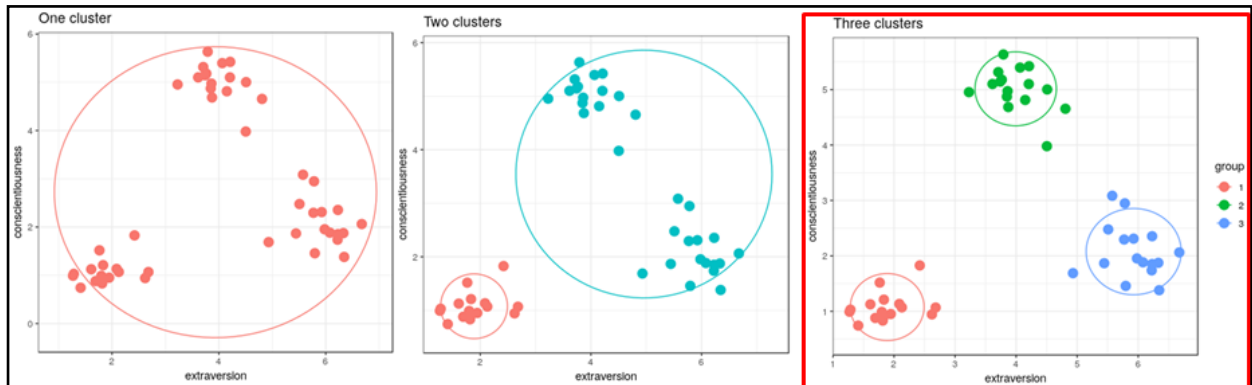
Example: Segregation of points learning objective

To satisfy the K-Means clustering segregation criteria, the clustering solution should have little overlap. We can see overlap (non-segregation) between two bottom-left clusters in the right solution, which violate segregation criteria. Thus the best solution is the middle one.



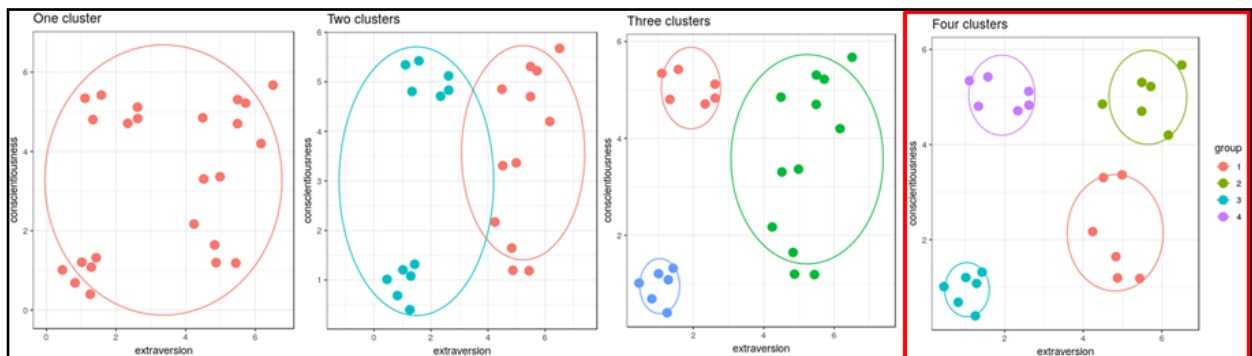
Example: Separation of points Learning objective.

There should be gaps between clusters (**separation; Between/Within SS**) to satisfy separation criteria. However, we can see too clear boundaries (separation) in the red cluster (the left solution) and in the blue cluster (the middle solution). Thus the best solution is the rightmost one in the Figure.



Module: Troubleshoot/Induce Error (Example: Simplicity learning objective)

The more clusters there are, we require stronger evidence (simplicity). There are no overlaps among all the clusters in these four clustering solutions. In addition, there are no clear boundaries among clusters. However, we also see separation, quite clear boundaries, in the first three solutions, which means that it may be reasonable to separate the clusters. To satisfy the simplicity example, we consider the four-cluster solution as the optimal one.



This tutorial is based on a misconception about simplicity we found a number of novices held: they were often overly biased toward simpler solutions with fewer clusters. Thus, they are likely to select a simpler less appropriate model in this case, and if they do, it provides an opportunity to illustrate the relative importance of simplicity versus other concerns

Module: Rule Induction

Verbal Rule Explanation (A): This concise verbal explanation should articulate the rule, how the rule works, and the AI system's outputs as per the rule. The verbal explanation should also

cover cases of where the AI system succeeds and where it fails. This should contain enough details for the learner to make their own judgments of the AI system's output.

The verbal explanation might also be a "non-rule". For example, there might be natural human tendencies to believe that a digit classifier will evaluate a 1 drawn with a curved line and classify it as another digit, such as a 5. A non-rule will explain the improbability of this classification. This also provides useful information, just as a rule does, but by deterring an individual from making what might appear to be a logical assumption of a system.

Visual Examples (B): This can be a visual representation of how the AI system succeeds and fails. It is helpful to have three examples of successes and three examples of failures. This will help individuals utilize the principle of good form, conceptualizing groups of what it looks like for a system to successfully classify vs what it looks like when it fails to classify correctly.








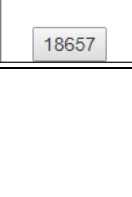
Depiction of Probabilistic Sensitivity of the Rule (C): This representation should contain some form of a 10x10 grid, with color-coded/text-labeled components that represent the Bayesian probabilities of the AI system's output. It should be a gauge of the sensitivity of the rule, by not only stating when the AI system will succeed or fail, but also having a frequency statement (Gigerenzer, 2003) of how likely the system will succeed/fail given a certain input. This is meant to push the system learner to make a decision one way or another; it is not on its own diagnostic.

Summary of Rule Effectiveness (D): This highlights the discriminability of the rule in both success and failure cases. It is meant to be helpful, but not its own diagnostic. It is a verbal explanation of the probabilities.

Practice

After the rule is stated, visualized and its sensitivity revealed, the learner will go through a practice session, where they will be presented with stimuli, and be asked to predict how the AI system will work. In this example, they will be shown a digit, and they will need to identify if the classifier will classify the digit as a 1 or as a 5. Once the learner makes a selection, they will be given immediate feedback, stating whether they were correct or incorrect, and why they were correct/incorrect. The correctness explanations will contain direct references of the rules.

Below is a table containing 8 practice samples. Each of the above referenced rules (i.e. flag/no flag, slanted left/right) is practiced two times, once in a correction classification (i.e. 1 classified as a 1) and once in an incorrect classification (i.e. 1 classified as a 5).

Practice #	Stimulus	Feedback
1		<p>“That’s correct, this is slanted to the right, so it classified it as a 1.”</p> <p>“That’s incorrect, this is slanted to the right, so it classified it as 1.”</p>
2		<p>“That’s correct, this doesn’t have a flag, so it classified it as a 1.”</p> <p>“That’s incorrect, this doesn’t have a flag, so it classified it as 1.”</p>
3		<p>“That’s correct, this has a flag, so it classified it as a 5.”</p> <p>“That’s incorrect, this has a flag, so it classified it as 5.”</p>
4		<p>“That’s correct, this is slanted to the left, so it classified it as a 5.”</p> <p>“That’s incorrect, this is slanted to the left, so it classified it as 5.”</p>
5		<p>“That’s correct, this is slanted to the right, so it classified it as a 1.”</p> <p>“That’s incorrect, this is slanted to the right, so it classified it as 1.”</p>
6		<p>“That’s correct, this is slanted to the left, so it classified it as a 5.”</p> <p>“That’s incorrect, this is slanted to the left, so it classified it as 5.”</p>
7		<p>“That’s correct, this doesn’t have a flag, so it classified it as a 1.”</p> <p>“That’s incorrect, this doesn’t have a flag, so it classified it as 1.”</p>
8		<p>“That’s correct, this has a flag, so it classified it as a 5.”</p> <p>“That’s incorrect, this has a flag, so it classified it as 5.”</p>

Module: Non-Rule Training

Verbal Rule Explanation (A): This concise verbal explanation should articulate the rule, how the rule works, and the AI system's outputs as per the rule. The verbal explanation should also cover cases of where the AI system succeeds and where it fails. This should contain enough details for the learner to make their own judgments of the AI system's output.

In this case, the verbal explanation is a "non-rule". For example, there might be natural human tendencies to believe that a digit classifier will evaluate a 3 drawn with a pigtail and classify it as a 2. A non-rule will explain the improbability of this classification. This also provides useful information, just as a rule does, but by deterring an individual from making what might appear to be a logical assumption of a system.

Visual Examples (B): This can be a visual representation of how the AI system succeeds and fails. It is helpful to have three examples of successes and three examples of failures. This will help individuals utilize the principle of good form, conceptualizing groups of what it looks like for a system to successfully classify vs what it looks like when it fails to classify correctly.





Depiction of Probabilistic Sensitivity of the Rule (C): This representation should contain some form of a 10x10 grid, with color-coded/text-labeled components that represent the Bayesian probabilities of the AI system's output. It should be a gauge of the insensitivity of the non-rule, by not only stating when the AI system will succeed or fail, but also having a frequency statement (Gigerenzer, 2003) of how likely the system will succeed/fail given a certain input. This is meant to push the system learner to make a decision one way or another; it is not on its own diagnostic.

Summary of Rule Effectiveness (D): This highlights the lack of discriminability of the non-rule in both success and failure cases. It is meant to be helpful, but not its own diagnostic. It is a verbal explanation of the probabilities.

Practice

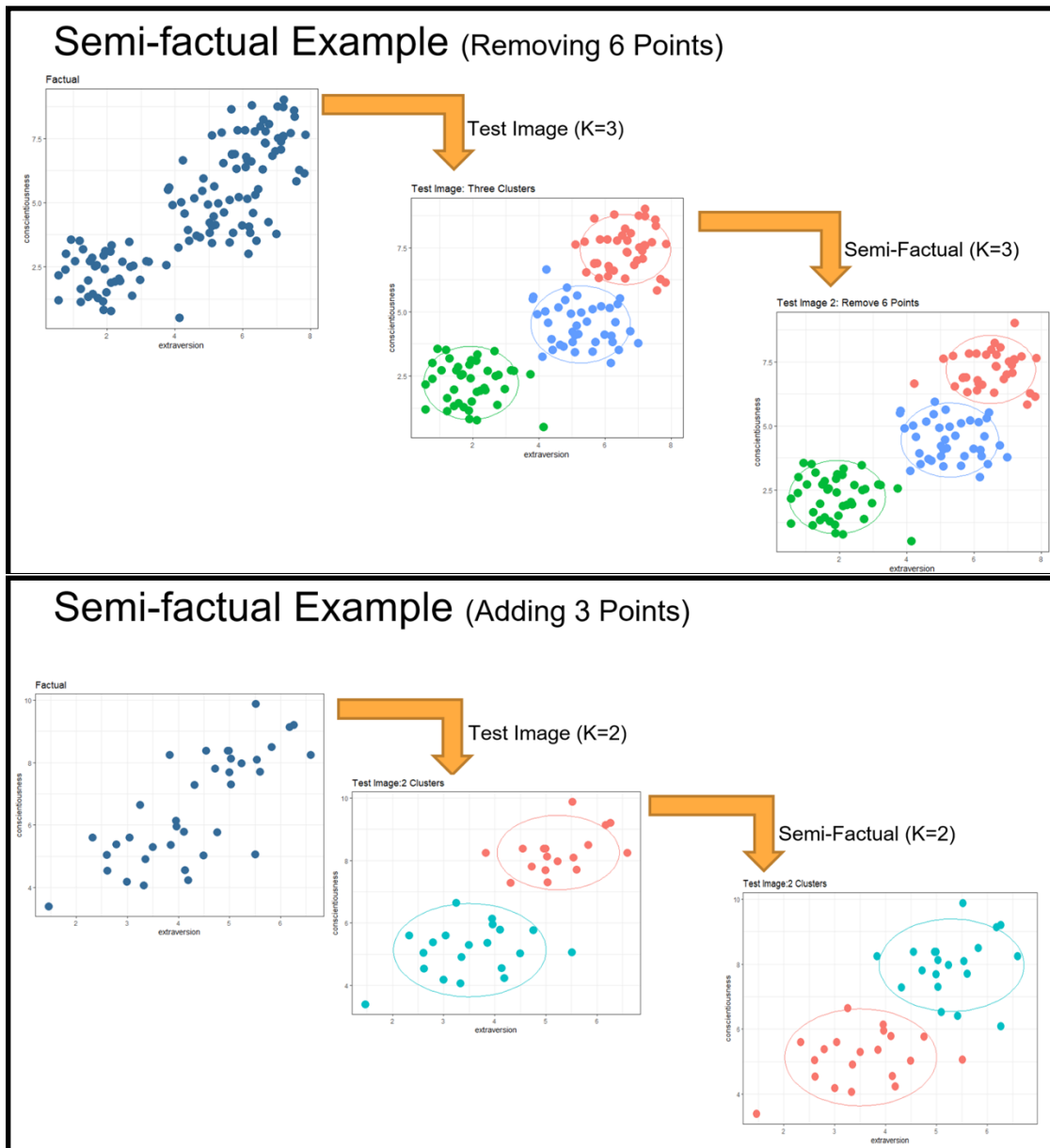
After the non-rule is stated, visualized and its sensitivity revealed, the learner will go through a practice session, where they will be presented with stimuli, and be asked to predict how the AI system will work. In this example, they will be shown a digit, and they will need to identify if the classifier will classify the digit as a 2 or as a 3. Once the learner makes a selection, they will be given immediate feedback, stating whether they were correct or incorrect, and why they were correct/incorrect. The correctness explanations will contain direct references of the non-rules.

Below is a table containing 4 practice samples. The above referenced rules (i.e. 3 with/without a pigtail) is practiced two times, once in a correct classification (i.e. 3 classified as a 3) and once in an incorrect classification (i.e. 3 classified as a 2).

Practice #	Stimulus	Feedback
1	 52752	“That’s correct, even though this has a pigtail, it classified it as a 3.” “That’s incorrect, even though this has a pigtail, it classified it as 3.”
2	 16030	“That’s correct, even though this doesn't have a pigtail, it classified it as a 3.” “That’s incorrect, even though this doesn't have a pigtail, it classified it as 3.”
3	 49986	“That’s correct, even though this does have a pigtail, it classified it as a 2.” “That’s incorrect, even though this does have a pigtail, it classified it as 2.”
4	 28654	“That’s correct, even though this doesn't have a pigtail, it classified it as a 2.” “That’s incorrect, even though this doesn't have a pigtail, it classified it as 2.”

Module: Semi-factual Example

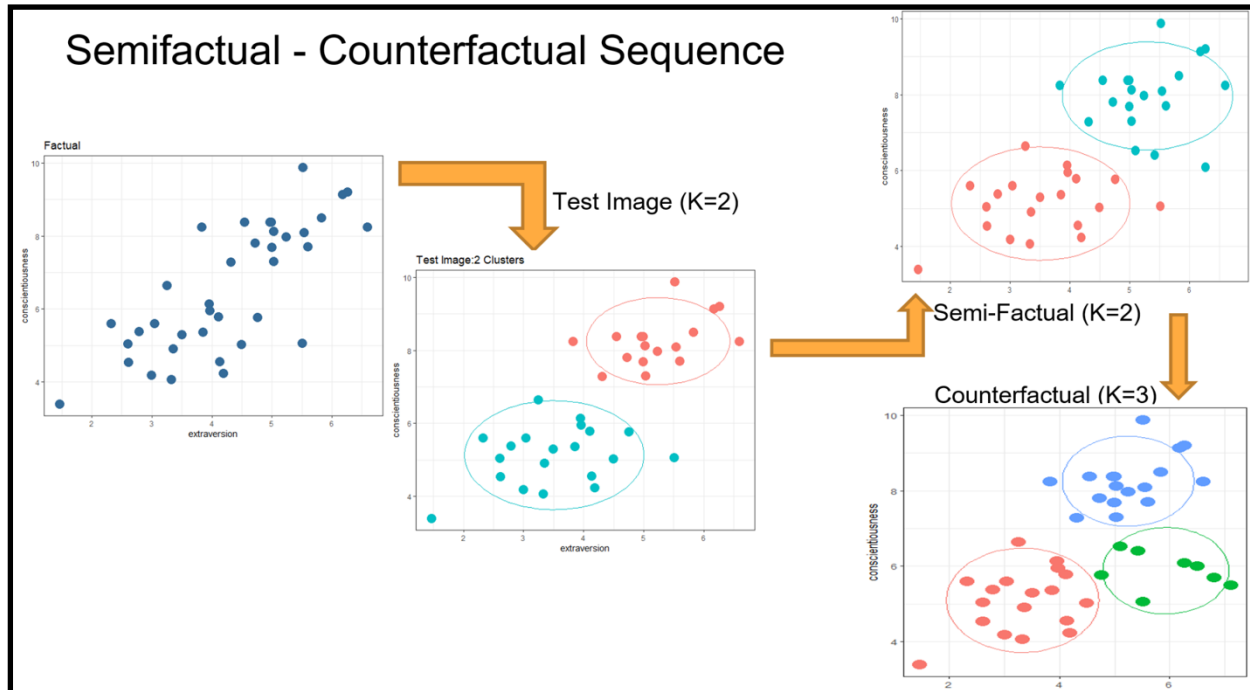
This module introduces semi-factual examples where rule-based training for a learner to learn K-Means clustering algorithm is not required. A learner can learn and visualize the clustering results without training in advance. In a semi-factual condition, K-Means clustering result remains the same, even if the modification involves removing points or adding points from the test image. (See the Figures below). This type of modification results in no change in classification as it delivers contrastive explanations without crossing a decision boundary, which can decrease the featural changes needed to generate the semi-factual explanation.



Module: Semi-factual - Counterfactual Sequence

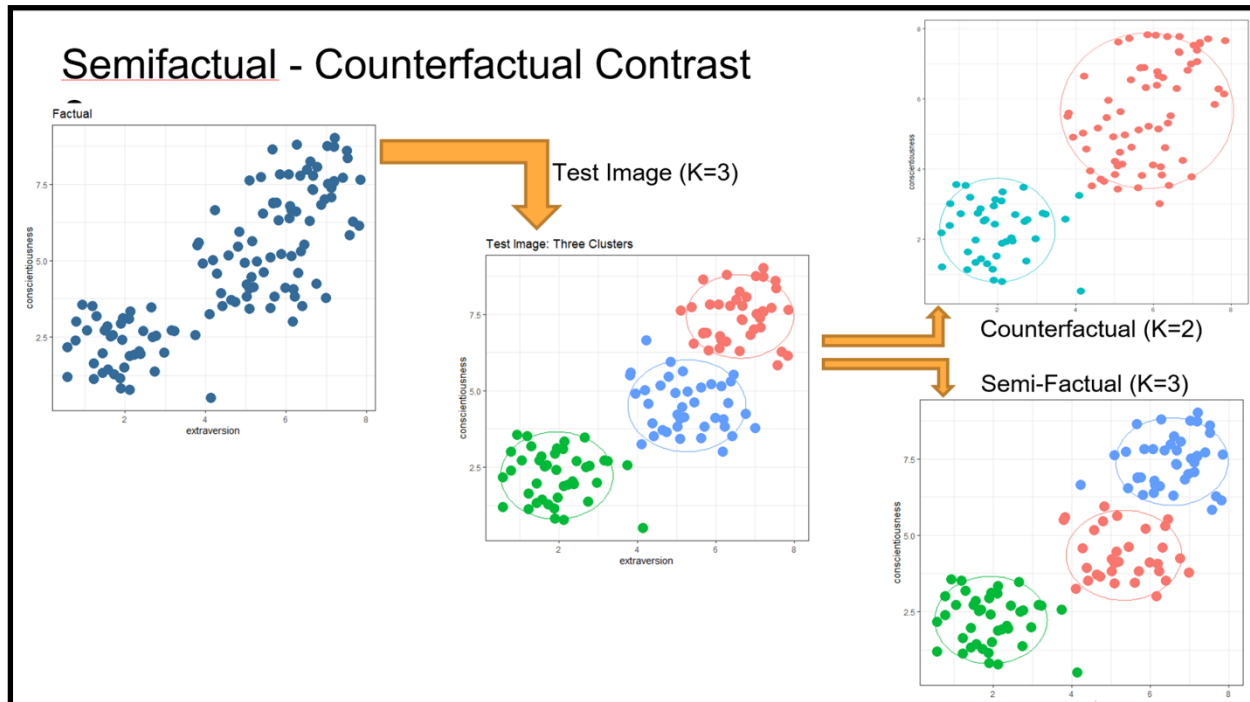
The focus of this module is to create an example where the explanation of the K-Mean clustering results should be “contrastive” to facilitate a user to learn about the to-be-explained algorithm rule. The example we created involves contrasting explanations which modify the current outcome of a clustering test image (K=2) to a semi-factual result (K=2), a not-changed-in-the-explanation result, and then to a counterfactual result (K=3), a changed-to-another-class result (see Figure below). This sequence is 1) we created a test image with two clusters; 2) we then created a semi-factual outcome where we added three points with an intact clustering result which does not require a change to the clustering explanation; 3) finally, we continued adding

three additional points resulting in this modification going beyond the decision boundary and the plausible clustering result changed from two to three clusters. This sequence helps us visualize the contrastive shift from semi-factual to counterfactual explanation using an experience-guided approach by identifying exceptional points added to the test image.



Module: Semi-factual - Counterfactual Contrast

This module consists of two examples which have used a method that modifies points/features to generate semi-factual and counterfactual explanations. In the examples, a test image changes to either semi-factual clustering result or counterfactual one with only one step, removing 10 points from the test image. The difference between these two examples is whether the modification goes beyond the decision boundary. A semi-factual example is constituted when we removed 10 points from the top-left of the test image ($K=3$) and the clustering result remains the same ($K=3$), whereas a contrastive, counterfactual example is created when we removed 10 points from the middle of the test image and the plausible clustering result goes beyond the decision boundary and changes to another clustering ($K=2$) (see Figure below).



Module: Semi-factual - Counterfactual Contrast

Verbal Rule Explanation (A): This concise verbal explanation should articulate the rule, how the rule works, and the AI system's outputs as per the rule. The verbal explanation should also cover cases of where the AI system succeeds and where it fails. This should contain enough details for the learner to make their own judgments of the AI system's output.

Visual Examples (B): This can be a visual representation of how the AI system succeeds and fails. It is helpful to have three examples of successes and three examples of failures. This will help individuals utilize the principle of good form, conceptualizing groups of what it looks like for a system to successfully classify vs what it looks like when it fails to classify correctly.

Depiction of Probabilistic Sensitivity of the Rule (C): This representation should contain some form of a 10x10 grid, with color-coded/text-labeled components that represent the Bayesian probabilities of the AI system's output. It should be a gauge of the sensitivity of the rule, by not only stating when the AI system will succeed or fail, but also having a frequency statement (Gigerenzer, 2003) of how likely the system will succeed/fail given a certain input. This is meant to push the system learner to make a decision one way or another; it is not on its own diagnostic.

Summary of Rule Effectiveness (D): This highlights the discriminability of the rule in both success and failure cases. It is meant to be helpful, but not its own diagnostic. It is a verbal explanation of the probabilities.

Module: Mental Model Matrix

We have not explored developing a full mental model matrix (MMM) tutorial, but have used the MMM for a similar application: evaluating the effectiveness of training material about calculus applied to a real-world modeling problem. Here, the MMM was employed to assess student self-reflection and self-explanation after they watched a video and tried to make sense of what they have learned from it. Participants were 2000-level (sophomores, n=34) and 4000-level (seniors, n=35) college students who majored in Mechanical Engineering at a university in the Midwest. They volunteered to participate in this learning activity to get extra bonus points for their homework grades. The Pandemic video, aiming at connecting calculus learning with real life situations, was originally created as a means to increase student motivation and interest in Calculus learning especially for beginners.

The purpose of using the MMM was for program evaluation, but we use some of the responses here to illustrate how this might be turned into a tutorial lesson. The steps include: 1) we combined all the matrix results into one matrix with four quadrants, which gives us a 69-page table; 2) we separated senior results with sophomore ones to obtain two group metrics, each with about 30 pages; 3) we then removed the overlapped and redundant student comments, which ends up in 8 pages in both groups; 4) we reviewed the trimmed metrics and then selected important comments, which synthesized to 2 pages in both groups; 5) finally, we analyzed the synthesized metrics and came up with a Commonality Matrix (see table below, with items commented by both student groups) and a Contrastive Matrix (see table below, with unique items commented by either senior or sophomore students only). These comments are illustrative of the kinds of comments that might be identified using such a process. For implementing a tutorial, these categories can be used to illustrate to new users the problems others have encountered, and the ways more experienced users understand different concepts.

Commonality Matrix of the Mental Model Matrix

Example statements regarding a video training system.

	Positive (+)	Negative (-)
Video/System	<p>How the video/system works: Components, connections, causal relationships, process control logic</p> <p>Utilization of current events into the material was a fantastic way of bringing relevance to the material.</p> <p>The logistic equation helps to account for more variables than the exponential equation and considers immunization when calculating the rate of infection. It will never be completely accurate because the demographic of each area is different, but it gives a good indicator of what areas should expect.</p>	<p>How the video/system fails: Breakdowns and limitations</p> <p>Being that this is true the video could include more example problems to allow the student to better understand modeling using differential equations. Having more than one work through problem could eliminate the limitation of students not being able to ask questions.</p> <p>The video could have utilized color coding in the equations to better distinguish between each step of the process.</p>
Yourself/Person	<p>How to make the video/system work: Detecting anomalies, performing workarounds and adaptations. What have you learned from this video?</p> <p>The model can be updated each day and help to move the curve so that it is more accurate.</p> <p>I learned about how spread equations are derived, as well as how they differ from a death curve.</p>	<p>How users (e.g., you) get confused: The kinds of errors people are likely to make. What would you like to learn/know more about this video?</p> <p>Users may not fully understand how to implement these functions right off the bat, meaning there could potentially be more worked-through examples to have the student gain practice and confidence in their work.</p> <p>I would like to learn more about how to refine the mathematical model presented in this video.</p>

Contrastive Matrix of the Mental Model Matrix

Comparison of feedback over different levels of experience (sophomore vs. senior).

	Positive (+)		Negative (-)	
Video/System	How the video/system works: Components, connections, causal relationships, process control logic		How the video/system fails: Breakdowns and limitations	
	Sophomore	Senior	Sophomore	Senior
	The logistic equation helps to account for more variables than the exponential equation and considers immunization when calculating the rate of infection. It will never be completely accurate because the demographic of each area is different, but it gives a good indicator of what areas should expect.	Tying in the trend to latency was a great idea, just requires more finesse in execution. Graphics and demonstration through herd immunity was incredibly well executed, and easy to follow!	Maybe talk about the different variations of the virus and how the strains are formed and what are the chances of that mutation of the virus happening.	This video fails in the fact that some students will not learn if it is just a teacher talking over lecture slides. It is important to show you work especially in math classes, so seeing a teacher work through the problem by hand is beneficial.
Yourself/Person	How to make the video/system work: Detecting anomalies, performing workarounds and adaptations. What have you learned from this video?		How users (e.g., you) get confused: The kinds of errors people are likely to make. What would you like to learn/know more about this video?	
	Sophomore	Senior	Sophomore	Senior
	I did not think to model the pandemic in this way at all before. Even being past beginners calc, there are not many examples in the program that are this real world. Obviously in other classes calc is used in great examples, but this being solely calculus based was eye opening.	I have learned the importance of proper assumptions for our mathematical model. The video beautifully bypasses effects of demographics, environment, weather, covid strain type, age limitations, immune system of people etc. and bases all these effects in the initial condition which assumes what the trend of spread would be like Italy.	One error that users might make when watching these videos is applying this model to similar situations. Also, if they do not have a good grip on the basics of calculus, such as IVP in the video, there is room to make mistakes. I like the content; it is relatable and engaging.	I would have liked to see more comparison between the model and actual data from NYC for the specified time period. This would provide further validation for the model and deepen the meaningfulness of the connection.

Module: Cheatsheet

A Cheatsheet module may provide visual examples of many rules or lessons. It can augment other modules or serve as an outline for a series of lessons.

K-Meaning Clustering Cheat Sheet

Clustering

Clustering is the classification of points into different groups or partitioning of points into clusters. Often proximity based on defined distance measure. A cluster is a set of points that are similar to one another, but dissimilar to points in other clusters.

Clustering Algorithms

Clustering is an unsupervised machine learning task. It automatically discovers natural group in data set. Unlike supervised learning, such as predictive modeling, clustering algorithms only interpret the input data set and find natural clusters in feature space.

K-Means Clustering

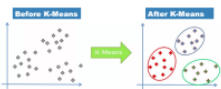
An unsupervised machine learning algorithm for clustering "n" points into "k" clusters where k is predefined constant

K-Mean Clustering Algorithm

The k-means clustering (Macqueen, 1967) algorithm is a data mining and machine learning tool used to cluster a set of points into groups without any prior knowledge of those relationships. It tries to identify if there are clusters of points in the data.

- Points in the same group are as similar as possible.
- Points in different group are as dissimilar as possible.

It cannot determine how many clusters exist. You tell it k, the number of clusters you want to use, and it gives you the best solution.



In the figure above, we see the data on the left. If we use k-means clustering with k=3, it finds the solution on the right.

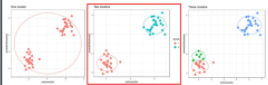
Selecting K: The Number of Clusters

The number of clusters is an INPUT to the k-means algorithm, not a output. Often, there is not one clear answer for what k should be. It is up to the analyst to decide which k is the best. Sometimes, more than one solution can be reasonable.

Criteria for Selecting K

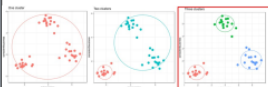
- The clusters should little overlap (**segregation**)
- There should be gaps between clusters (**separation; between/within SS**)
- The more clusters there are, we require stronger evidence (**simplicity**)

Rule of Segregation



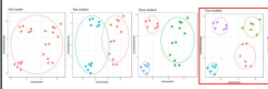
In the figure above, we can see overlap (non-segregation) between two bottom-left clusters in the right solution, which violate segregation criteria. Thus, the best solution is the middle one.

Rule of Separation



In the figure above, we can see too clear boundaries (separation) in the red cluster (the left solution) and in the blue cluster (the middle solution). Thus the best solution is the right one.

Rule of Simplicity

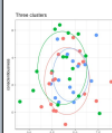


In the figure above, we can see separation, quite clear boundaries, in the first three solutions, which means that it may be reasonable to select four-cluster solution.

How the Clustering Algorithm Works

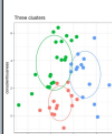
- The algorithm starts by randomly assigning each group to a cluster, and calculating the center of each cluster, called the centroid.
- Next, each data points is re-assigned to its nearest centroid, also called "cluster assignment step".
- After the assignment step, the algorithm computes the new mean value of each cluster, called "centroid update".
- Repeat until data points stay in the same cluster and the values of the centroids stabilize.

Example: Round 1



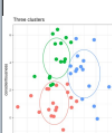
We start with completely random assignment to three groups. The three groups overlap, and are not separated.

Example: Round 2



Now, each point has been re-assigned to the center that it is closest to. The three groups are also redefined, and a new center is calculated.

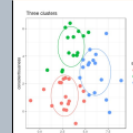
Example: Round 3



Generally, after the first assignment round, only a few points change on each round. The clusters adjust a little bit in the third round.

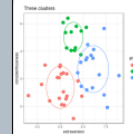
How the Clustering Algorithm Works (cont.)

Example: Round 4



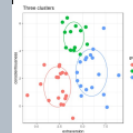
Now, only a few points change.

Example: Round 5



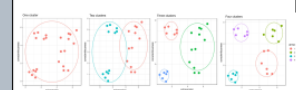
Again, a few more points change.

Example: Round 6



No points have moved since the last round. This is the final solution.

Review: How Many Ks (Clusters) to Choose?



The Optimal Solution (Four Ks (clusters) result):

- We can see no overlap among green, red, blue, and purple clusters.
- We see clear boundaries among four clusters.
- We also see clear boundary (separation) between green and red clusters (these two clusters were one cluster in three Ks solution).
- This shows an optimal result.

References

- Borders, J., Klein, G., & Besuijen, R. (2019). An operational account of mental models: A pilot study. International Conference on Naturalistic Decision Making, San Francisco, CA.
- Crandall, B., Klein, G., and Hoffman R. R. (2006). *Working Minds: A Practitioner's Guide to Cognitive Task Analysis*. Cambridge, MA: MIT Press.
- deWinter, J. (2016). Just playing around: From procedural manuals to in-game training. In *Computer Games and Technical Communication* (pp. 89-106). Routledge.
- Dorsel, T. N., & Cundiff, G. W. (1979). The cheat-sheet: Efficient coding device or indispensable crutch?. *The Journal of Experimental Education*, 48(1), 39-42.
- Gigerenzer, G. (2003). Why does framing influence judgment?. *Journal of General Internal Medicine*, 18(11), 960.
- Halverson, C. A., & Ackerman, M. S. (2008). The birth of an organizational resource: The surprising life of a cheat sheet. In *Resources, Co-Evolution and Artifacts* (pp. 9-35). Springer, London.
- Kenny, E.M., & Keane, M.T. (2020). On generating plausible counterfactual and semi-factual explanations for deep learning. *arXiv preprint arXiv:2009.06399*.
- Klein, G., & Borders, J. (2016). The ShadowBox approach to cognitive skills training: An empirical evaluation. *Journal of Cognitive Engineering and Decision Making*, 10(3), 268-280.
- Klein, G., Hintze, N., & Saab, D. (2013, May). Thinking inside the box: The ShadowBox method for cognitive skill development. In *Proceedings of the 11th International Conference on Naturalistic Decision Making* (pp. 121-124).
- Koopman, P., & Hoffman, R. R. (2003). Work-arounds, make-work, and kludges. *IEEE Intelligent Systems*, 18(6), 70-75.
- Metcalf, J. (2017). Learning from errors. *Annual review of psychology*, 68, 465-489.
- Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267, 1-38.
- Mueller, S. T., & Klein, G. (2011). Improving users' mental models of intelligent software tools. *IEEE Intelligent Systems*, 26(2), 77-83.
- Mueller, S. T., Klein, G., & Burns, C (2009). Experiencing the tool without experiencing the pain: Concepts for an experiential user guide. In *Proceedings of the Ninth International Conference on Naturalistic Decision Making*, London, UK, June 2009.
- Mueller, S. T., Veinott, E. S., Hoffman, R. R., Klein, G., Alam, L., Mamun, T., & Clancey, W. J. (2021). Principles of Explanation in Human-AI Systems. *arXiv preprint arXiv:2102.04972*.
- Nushi, B., Kamar, E., Horvitz, E., & Kossmann, D. (2017). On human intellect and machine failures: Troubleshooting integrative machine learning systems. In *Thirty-First AAAI Conference on Artificial Intelligence*.
- Nushi, B., Kamar, E., & Horvitz, E. (2018). Towards accountable ai: Hybrid human-machine analyses for characterizing system failure. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* (Vol. 6, No. 1).
- Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.