Machine learning is ubiquitous: detecting spam e-mails, flagging fraudulent purchases, and providing the next movie in a Netflix binge. But few users at the mercy of machine learning *outputs* know what's happening behind the curtain. My research goal is to demystify the black box for non-experts by creating *algorithms that can inform, collaborate with, compete with, and understand users* in real-world settings.

This is at odds with mainstream machine learning—take topic models. Topic models are sold as a tool for understanding large data collections: lawyers scouring Enron e-mails for a smoking gun, journalists making sense of Wikileaks, or humanists characterizing the oeuvre of Lope de Vega. But topic models' proponents never asked what those lawyers, journalists, or humanists needed. Instead, they optimized *held-out likelihood*. When my colleagues and I developed the *interpretability* measure to assess whether topic models' users understood their outputs, we found that interpretability and held-out likelihood were negatively correlated! The topic modeling community (including me) had fetishized complexity at the expense of usability.

Since this humbling discovery, I've built topic models that are a collaboration between humans and computers. The computer starts by proposing an organization of the data. The user responds by separating confusing clusters, joining similar clusters together, or comparing notes with another user. The model updates and then directs the user to problematic areas that it knows are wrong. This is a huge improvement over the "take it or leave it" philosophy of most machine learning algorithms.

This is not only a technical improvement but also an improvement to the social process of machine learning adoption. A program manager who used topic models to characterize NIH investments uncovered interesting synergies and trends, but the results were unpresentable because of a fatal flaw: one of the 700 clusters lumped urology together with the nervous system, anathema to NIH insiders. Our tools allow non-experts to fix such obvious (to a human) problems, allowing machine learning algorithms to overcome the *social* barriers that often hamper adoption.

Our realization that humans have a lot to teach machines led us to *simultaneous machine interpretation*. Because verbs end phrases in many languages, such as German and Japanese, existing algorithms must wait until the end of a sentence to begin translating (since English sentences have verbs near the start). We learned tricks from professional human interpreters—passivizing sentences and guessing the verb—to translate sentences sooner, letting speakers and algorithms cooperate together and enabling more natural cross-cultural communication.

The reverse of cooperation is competition; it also has much to teach computers. I've increasingly looked at language-based games whose clear goals and intrinsic fun speed research progress. For example, in *Diplomacy*, users chat with each other while marshaling armies for world conquest. Alliances are fluid: friends are betrayed and enemies embraced as the game develops. However, users' conversations hint when friendships break: betrayers writing ostensibly friendly messages become more polite, stop talking about the future, and change how much they write. Diplomacy may be a nerdy game, but it is a fruitful testbed to teach computers to understand messy, emotional human interactions.

A game with higher stakes is politics. However, just like Diplomacy, the words that people use reveal their underlying goals; computational methods can help expose the "moves" political players can use. With collaborators in political science, we've built models that: show when politicians in debates strategically change the topic to influence others; frame topics to reflect political leanings; use subtle linguistic phrasing to express their political leaning; or create political subgroups with larger political movements.

Conversely, games also teach humans *how computers think*. Our trivia-playing robot played four former Jeopardy champions in front of 600 high school students. The computer's early lead evaporated because we foolishly projected the computer's thought process for all to see. Our opponents learned to read the algorithm's ranked dot products and adjusted their strategy. In five years of teaching machine learning, students never so quickly learned how linear classifiers work. The probing questions from high school students in the audience showed they understood too. (Later, we played again against Ken Jennings; he sat in front of the dot products and our system won.)

Advancing machine learning requires closer, more natural interactions. However, we still require much of the user—reading distributions or dot products—rather than natural language interactions. Document exploration tools should describe in words what a cluster is, not just provide inscrutable word clouds; deception detection systems should say *why* a betrayal is imminent; and question answers should explain *how* it knows Aaron Burr shot Alexander Hamilton. My work will complement machine learning's ubiquity with transparent, empathetic, and useful interactions with users.