User Needs + Defining Success

Even the best AI will fail if it doesn't provide unique value to users.

This chapter covers:

- Which user problems is AI uniquely positioned to solve?
- How can we augment human capabilities in addition to automating tasks?
- How can we ensure our reward function optimizes AI for the right thing?

Want to drive discussions, speed iteration, and avoid pitfalls? Use the worksheet.

What’s new when working with AI

When building any product in a human-centered way, the most important decisions you’ll make are: Who are your users? What are their values? Which problem should you solve for them? How will you solve that problem? How will you know when the experience is “done”?

In this chapter, we’ll help you understand which user problems are good candidates for AI and how to define success. Key considerations:

① **Find the intersection of user needs & AI strengths**. Solve a real problem in ways in which AI adds unique value.

② **Assess automation vs. augmentation**. Automate tasks that are difficult, unpleasant, or where there’s a need for scale; and ideally ones where people who currently do it can agree on the “correct” way to do it. Augment tasks that people enjoy doing, that carry social capital, or where people don’t agree on the “correct” way to do it.

③ **Design & evaluate the reward function**. The “reward function” is how an AI defines successes and failures. Deliberately design this function with a cross-functional team, optimizing for long-term user benefits by imagining the downstream effects of your product. Share this function with users when possible.

① Find the intersection of user needs & AI strengths

Like any human-centered design process, the time you spend identifying the right problem to solve is some of the most important in the entire effort. Talking to people, looking through data, and observing behaviors can shift your thinking from technology-first to people-first.

The first step is to identify real problems that people need help with. There are many ways to discover these problems and existing resources online to help you get started. We recommend looking through IDEO's Design Kit methods section for examples of how to find problems and corresponding user needs.

For our example app RUN, user needs might be:

- The user wants to get more variety in their runs so they don't get bored and quit running.
- The user wants to track their daily runs so that they can get ready for a 10k in six months.
- The user would like to meet other runners at their skill level so they can stay motivated to keep running.

You should always build and use AI in responsible ways. When deciding on which problem to solve, take a look at the Google AI Principles, and Responsible AI Practices for practical steps, to ensure you're building with the greater good in mind. For starters, make sure to get input from a diverse set of users early on in your product development process. Hearing from many different points of view can help you avoid missing out on major market opportunities or creating designs that unintentionally exclude specific user groups.

Map existing workflows

Mapping the existing workflow for accomplishing a task can be a great way to find opportunities for AI to improve the experience. As you walk through how people currently complete a process, you'll better understand the necessary steps and identify aspects that could be automated or augmented. If you already have a working AI-powered product, test your assumptions with user research. Try letting people use your product (or a "Wizard of Oz" test) to automate certain aspects of the process, and see how they feel about the results.

Decide if AI adds unique value

Once you identify the aspect you want to improve, you'll need to determine which of the possible solutions require AI, which are meaningfully enhanced by AI, and which solutions don't benefit from AI or are even degraded by it.

It's important to question whether adding AI to your product will improve it. Often a rule or heuristic-based solution will work just as well, if not better, than an AI version. A simpler solution has the added benefit of being easier to build, explain, debug, and maintain. Take time to critically consider how introducing AI to your product might improve, or regress, your user experience.

To get you started, here are some situations where an AI approach is probably better than a rule-based approach, and some in which it is not.

When AI is probably better

- **Recommending different content to different users**. Such as providing personalized suggestions for movies to watch.
- **Prediction of future events**. For example, showing flight prices for a trip to Denver in late November.
- **Personalization improves the user experience**. Personalizing automated home thermostats makes homes more comfortable and the thermostats more efficient over time.
- **Natural language understanding**. Dictation software requires AI to function well for different languages and speech styles.
- **Recognition of an entire class of entities**. It's not possible to program every single face into a photo tagging app — it uses AI to recognize two photos as the same person.
- **Detection of low occurrence events that change over time**. Credit card fraud is constantly evolving and happens infrequently to individuals, but frequently across a large group. AI can learn these evolving patterns and detect new kinds of fraud as they emerge.
- **An agent or bot experience for a particular domain**. Booking a hotel follows a similar pattern for a large number of users and can be automated to expedite the process.

- **Showing dynamic content is more efficient than a predictable interface**. AI-generated suggestions from a streaming service surface new content that would be nearly impossible for a user to find otherwise.

When AI is probably not better

- **Maintaining predictability**. Sometimes the most valuable part of the core experience is its predictability, regardless of context or additional user input. For example, a “Home” or “Cancel” button is easier to use as an escape hatch when it stays in the same place.

- **Providing static or limited information**. For example, a credit card entry form is simple, standard, and doesn’t have highly varied information requirements for different users.

- **Minimizing costly errors**. If the cost of errors is very high and outweighs the benefits of a small increase in success rate, such as a navigation guide that suggests an off-road route to save a few seconds of travel time.

- **Complete transparency**. If users, customers, or developers need to understand precisely everything that happens in the code, like with Open Source Software. AI can’t always deliver that level of explainability.

- **Optimizing for high speed and low cost**. If speed of development and getting to market first is more important than anything else to the business, including the value that adding AI would provide.

- **Automating high-value tasks**. If people explicitly tell you they don’t want a task automated or augmented with AI, that’s a good task not to try to disrupt. We’ll talk more about how people value certain types of tasks below.

Key concept

Instead of asking “Can we use AI to ________?”, start exploring human-centered AI solutions by asking:

- How might we solve _________ ?
- Can AI solve this problem in a unique way?

Apply the concepts from this section in Exercise 1 in the worksheet

② Assess automation vs. augmentation

When you've found the problem you want to solve and have decided that using AI is the right approach, you'll then evaluate the different ways AI can solve the problem and help users accomplish their goals. One large consideration is if you should use AI to automate a task or to augment a person's ability to do that task themselves.

Some tasks, people would love for AI to handle, but there are many activities that people want to do themselves. In those latter cases, AI can help them perform the same tasks, but faster, more efficiently, or sometimes even more creatively. When done right, automation and augmentation work together to both simplify and improve the outcome of a long, complicated process.

When to automate

Automation is typically preferred when it allows people to avoid undesirable tasks entirely or when the time, money, or effort investment isn't worth it to them. These are usually tasks people are happy to delegate to AI as they don't require oversight, or they can be done just as well by someone (or something) else. Successful automation is often measured by the following:

- Increased efficiency
- Improved human safety
- Reduction of tedious tasks
- Enabling new experiences that weren't possible without automation

Automation is often the best option for tasks that supplement human weaknesses with AI strengths. For example, it would take a human a very long time to sort through their photo library and group pictures by subject. AI can do that quickly and easily, without constant feedback. Consider automating experiences when:

People lack the knowledge or ability to do the task

There are many times when people would do something if they knew how, but they don't so they can't. Or they technically know how, but a machine is much better suited to the task — such as searching thousands of rows in a spreadsheet to find particular value.

There are also often temporary limitations on people, like needing to complete a task quickly, that can lead to preferences for giving up control. For example, one might save time by using the automated setting on their rice cooker when they're rushed to make dinner during the week, but make their sushi rice by hand over the weekend.

Tasks are boring, repetitive, awkward, or dangerous

There's little value in attempting to edit a document you wrote without using spell-check. It's unwise to check for a gas leak in a building using your own nose when you could use a sensor to detect the leak. In both of these situations, most people would prefer to give up control to avoid tasks that don't provide them value.

Even when you choose to automate a task, there should almost always be an option for human oversight — sometimes called "human in the loop" — and intervention if necessary. Easy options for this are allowing users to preview, test, edit, or undo any functions that your AI automates.

When to augment

When building AI-powered products, it's tempting to assume that the best thing you can do for your users is automate tasks they currently have to do manually. However, there are plenty of situations where people typically prefer for AI to augment their existing abilities and give them "superpowers" instead of automating a task away entirely.

Successful augmentation is often measured by the following:

- Increased user enjoyment of a task
- Higher levels of user control over automation
- Greater user responsibility and fulfillment
- Increased ability for user to scale their efforts
- Increased creativity

Augmentation opportunities aren't always easy to define as separate from automation, but they're usually more complicated, inherently human, and personally valuable. For example, you may use tools that automate part of designing a t-shirt, like resizing your art or finding compatible colors. The design software, in this case, augments the task of t-shirt design, and unlocks limitless silliness and ingenuity. Consider augmenting people's existing abilities when:

People enjoy the task

Not every task is a chore. If you enjoy writing music, you probably wouldn't want an AI to write entire pieces for you. If an algorithm did it for you, you won't get to participate in the creative process you love. However, using something like Magenta Studio could help you during the creative process without taking away the essential humanity of your artistic process.

Personal responsibility for the outcome is required or important

People exchange small favors all the time. Part of doing a favor for someone is the social capital you gain by giving up your time and energy. For tasks like these, people prefer to remain in control and responsible to fulfill the social obligations they take on. Other times, when there is no personal obligation, like paying tolls on roads, an automated system is typically preferred.

The stakes of the situation are high

People often want to, or have to, remain in control when the stakes are high for their role; for example pilots, doctors, or police officers. These can be physical stakes like ensuring someone gets off a tall ladder safely, emotional stakes like telling loved ones how you feel about them, or financial stakes like sharing credit card or banking information. Additionally, sometimes personal responsibility for a task is legally required. In low stakes situations, like getting song recommendations from a streaming service, people will often give up control because the prospect of discovery is more important than the low cost of error.

Specific preferences are hard to communicate

Sometimes people have a vision for how they want something done: a room decorated, a party planned, or a product designed. They can see it in their mind's eye but can't seem to do it justice in words. In these kinds of situations, people prefer staying in control so they can see their vision through. When people don't have a vision or don't have time to invest in one, they are more likely to prefer automation.

Key concept

Below are some example research questions you can ask to learn about how your users think about automation and augmentation:

- If you were helping to train a new coworker for a similar role, what would be the most important tasks you would teach them first?
- Tell me more about that action you just took, about how often do you do that?
- If you had a human assistant to work with on this task, what, if any duties would you give them to carry out?

Apply the concepts from this section in Exercise 2 in the worksheet

③ Design & evaluate the reward function

Any AI model you build or incorporate into your product is guided by a reward function, also called an “objective function”, or “loss function.” This is a mathematical formula, or set of formulas, that the AI model uses to determine “right” vs. “wrong” predictions. It determines the action or behavior your system will try to optimize for, and will be a major driver of the final user experience.

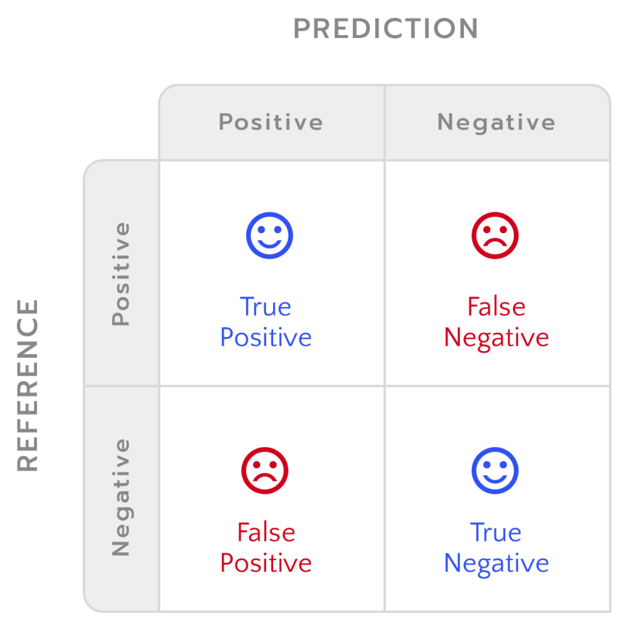
When designing your reward function, you must make a few key decisions that will dramatically affect the final experience for your users. We’ll cover those next, but remember that designing your reward function should be a collaborative process across disciplines. Your conversations should include UX, Product, and Engineering perspectives at the minimum. Throughout the process, spend time thinking about the possible outcomes, and bounce your ideas off other people. That will help reveal pitfalls where the reward function could optimize for the wrong outcomes.

Weigh false positives & negatives

Many AI models predict whether or not a given object or entity belongs to a certain category. These kind of models are called “binary classifiers”. We’ll use them as a simple example for understanding how AI’s can be right or wrong.

When binary classifiers make predictions, there are four possible outcomes:

- True positives . When the model correctly predicts a positive outcome.
- True negatives . When the model correctly predicts a negative outcome.
- False positives . When the model incorrectly predicts a positive outcome.
- False negatives . When the model incorrectly predicts a negative outcome.



A generic confusion matrix illustrating the two kinds of successes — true positives and true negatives — and two kinds of errors — false positives and false negatives — any AI model can make.

Let's walk through an example using our example app, RUN. Suppose RUN uses an AI model to recommend runs to users. Here's how the different model outcomes would play out:

1. True positives . The model suggested a run the user liked and chose to go on.
2. True negatives . The model did not suggest a run the user would not have chosen to go on.
3. False positives . The model suggested a run to the user that they did not want to go on.
4. False negatives . The model did not suggest a run to the user that they would have wanted to go on if they knew about it.

When defining the reward function, you'll be able to "weigh" outcomes differently. Weighing the cost of false positives and false negatives is a critical decision that will shape your users' experiences. It is tempting to weigh both equally by default. However, that's not likely to match the consequences in real life for users. For example, is a false alarm worse than one that doesn't go off when there's a fire? Both are incorrect, but one is much more dangerous. On the other hand, occasionally recommending a song that a person doesn't like isn't as lethal. They can just decide to skip it. You can mitigate the negative effects of these types of errors by including confidence indicators for a certain output or result.

Consider precision & recall tradeoffs

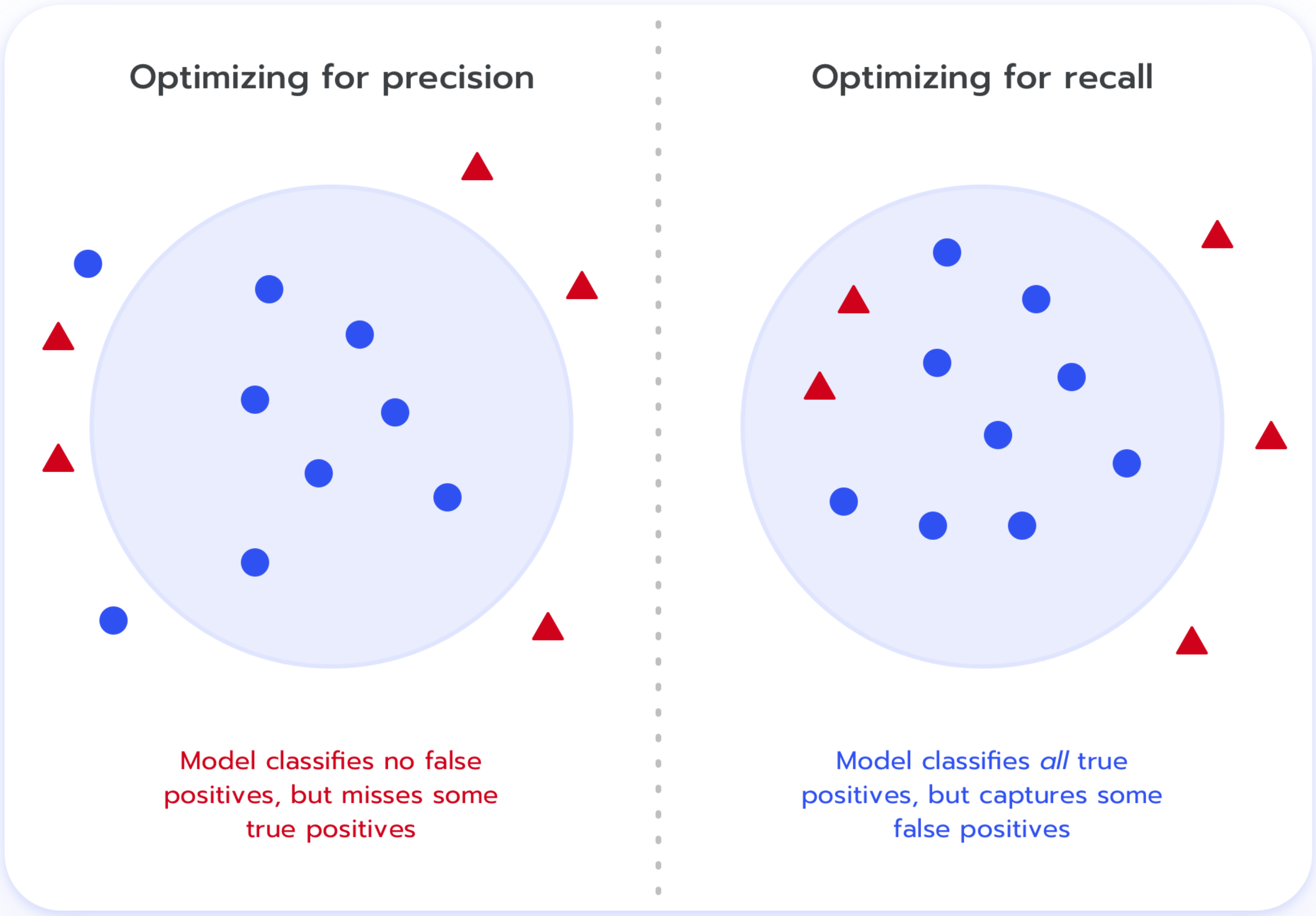
Precision and recall are the terms that describe the breadth and depth of results that your AI provides to users, and the types of errors that users see.

1. Precision refers to **the proportion of true positives correctly categorized out of all the true and false positives**.The higher the precision, the more confident you can be that any model output is correct. However, the tradeoff is that you will increase the number of false negatives by excluding possibly relevant results.

   For example, if the model above was optimized for precision, it wouldn't recommend every single run that a user might choose to go on, but it would be highly confident that every run it did recommend would be accepted by the user. Users would see very few if any runs that didn't match their preferences, but they might see fewer suggestions overall.

2. Recall refers to the **proportion of true positives correctly categorized out of all the true positives and false negatives**. The higher the recall, the more confident you can be that all the relevant results are included somewhere in the output. However, the tradeoff is that you will increase the number of false positives by including possibly irrelevant results.

   You can think of this as the model recommonding every run a user might want to go on, and including other runs the user does not choose to go on. The user however, would always have suggestions for a run, even if those runs didn't match their preferences as well.


Optimizing for precision
Optimizing for recall
Model classifies no false positives, but misses some true positives
Model classifies *all* true positives, but captures some false positives

A diagram showing the trade-offs when optimizing for precision or recall. On the left, optimizing for precision can reduce the number of false positives but may increase the number of false negatives. On the right, optimizing for recall catches more true positives but also increases the number of false positives.

You'll need to design specifically for these tradeoffs — there's no getting around them. Where along that spectrum your product falls should be based on what your users expect and what gives them the sense of task completeness. Sometimes, seeing some lower confidence results in addition to all of the 100% results can help users trust that the system isn't missing anything. In other cases, showing lower confidence results could lead to users trusting the system less. Make sure to test the balance between precision and recall with your users.

Evaluate the reward function outcomes

The next step is evaluating your reward function. Like any definition of success, it will be tempting to make it very simple, narrow, and immediate. However, this isn't the best approach: when you apply a simple, narrow, and immediate reward function to broad audiences over time, there can be negative effects.

Here are a few considerations when evaluating your reward function:

Assess inclusivity

You'll want to make sure your reward function produces a great experience for all of your users. Being inclusive means taking the time to understand who is using your product and making sure the user experience is equitable for people from a variety of backgrounds and perspectives, and across dimensions such as race, gender, age, or body shape, among many others. Designing AI with fairness in mind from the beginning is an essential step toward building inclusively. Open source tools like Facets and the What-If Tool allow you to inspect your datasets for potential bias. There's more on this in the Data Collection + Evaluation chapter.

While the Guidebook provides some advice related to fairness, it is not an exhaustive resource on the topic. Addressing fairness in AI is an active area of research. See Google's Responsible AI Practices for our latest fairness guidance and recommended practices.

Monitor over time

You'll also want to consider the implications of your chosen reward function over time. Optimizing for something like number of shares may seem like a good idea in the short term but over enough time, bombarding users with sharing notifications could create a very noisy experience. Imagine the best individual and collective user experience on their 100th or 1,000th day using your product as well as the first. Which behaviors and experiences should you optimize for in the long run?

Imagine potential pitfalls

Second-order effects are the consequences of the consequences of a certain action. These are notoriously difficult to predict but it’s still worth your time to consider them when designing your reward function. One useful question to ask is, “What would happen to our users/their friends & family/greater society if the reward function were perfectly optimized?” The result should be good. For example, optimizing to take people to the best web page for a search query is good, if it’s perfect. Optimizing to keep people’s attention continuously throughout the day may not provide them benefits in the long run.

Account for negative impact

As AI moves into higher stakes applications and use-cases, it becomes even more important to plan for and monitor negative impacts of your product’s decisions. Even if you complete all of the thought exercises in the worksheets, you probably won’t uncover every potential pitfall upfront. Instead, schedule a regular cadence for checking your impact metrics, and identifying additional potential bad outcomes and metrics to track.

It’s also useful to connect potential negative outcomes with changes to the user experience you could make to address them. For example, you could set the following standards and guidance for you and your team:

- If users’ average rate of rejection of smart playlists and routes goes above 20%, we should **check our ML model**.
- If over 60% of users download our app and never use it, we should **revisit our marketing strategy**.
- If users are opening the app frequently, but only completing runs 25% of the time, we’ll talk to users about their experiences and potentially **revisit our notification frequency**.

As your product matures, check in your product feedback for negative impacts on stakeholders you didn’t consider. If you find some stakeholders are experiencing negative effects on account of your product, talk to them to understand their situation. Based on these conversations, strategize ways to adapt your product to avoid continued negative impact.

An easy way to keep an eye on negative impacts is through social media or alert systems like Google Alerts. Make sure you’re listening to your users and identifying potential unintended consequences as early as possible.

Key concept

Everyone on your team should feel aligned on what both success and failure look like for your feature, and how to alert the team if something goes wrong. Here's an example framework for this:

> If **{ specific success metric }** for __ { your team's AI-driven feature } { drops below/goes above }__ { meaningful threshold } We will **{ take a specific action }**.

Apply the concepts from this section in Exercise 4 in the worksheet

Summary

Aligning your product with user needs is step one in any successful AI product. Once you've found a need, you should evaluate whether using AI will uniquely address the need. From there, consider whether some parts of the experience should be automated or augmented. Lastly, design your reward function to create a great user experience for all your users over the long run.

① **Find the intersection of user needs & AI strengths**. Make sure you're solving a real problem in a way where AI is adding unique value. When deciding on which problem to solve, you should always build and use AI in responsible ways. Take a look at the Google AI Principles and Responsible AI Practices for practical steps to ensure you are building with the greater good in mind.

② **Assess automation vs. augmentation**. Automate tasks that are difficult or unpleasant, and ideally ones where people who do it currently can agree on the "correct" way to do it. Augment bigger processes that people enjoy doing or that carry social value.

③ **Design & evaluate the reward function**. The "reward function" is how an AI defines successes and failures. You'll want to deliberately design this function including optimizing for long-term user benefits by imagining the downstream effects of your product and limiting their potentially negative outcomes.

Want to drive discussions, speed iteration, and avoid pitfalls? Use the worksheet