Unbox

The debugging workspace for machine learning

Unbox

Table of Contents

Chapter 1

On performance and trust

Machine learning (ML) models have been rapidly gaining space and relevance in our society. They promise to disrupt numerous application areas by transforming decision-making and human-computer interaction. **Despite the recent astronomical advances in the practical front, most of the models are still seen as black-boxes by their creators**. This means that even though they are used routinely, their true inner workings and the full extent of their behaviors remain a mystery.

Some models, when deployed, shall directly affect the lives of billions of people in the most diverse situations, from quickly triaging medical images in a crowded hospital to influencing how you spend your hard-earned money online.

Why should you trust my model performance?

The question, then, is: **if we do not truly understand what's going on under the hood for these ML models, can we even trust them?** Should they be deployed in production?

Most people don't have a choice and the encounters with a model that is not working correctly (or worse – is exhibiting biases and behaving in unethical ways) can be unpleasant at best and tragic at worst. As a consequence, not only the users and the companies suffer: the machine learning community as a whole takes a toll. It is impossible to develop the tools that indeed have the potential to propel humanity forward in many ways without society's trust.

Aggregate metrics are not enough

If the consequences of deploying ML models without truly understanding them are dire, **then why are ML engineers and data scientists doing it every day?**

The truth is, most of them also didn't have a clue their models would behave the way they did. Today, a lot of machine learning practice revolves around optimizing over aggregate metrics*. In industry, this often means shipping the model with the highest accuracy, and in academia, it corresponds to striving to beat the state-of-the-art on a benchmark.

The problem is that **aggregate metrics are not enough**. They do not tell the whole story about model performance and, in fact, provide a distorted view of its quality.

> A 90% accuracy obtained via cross-validation might look good on a landing page or on an academic paper, **but it tells little about what the model has actually learned or how that accuracy translates to different subsets of the data**. Furthermore, such a metric shows only a glimpse of how the model will behave in the wild, where it will encounter a long, long tail of edge cases.

For example, a fraud classifier that always predicts the majority class would have very high accuracy, but that model is useless. Even by changing the aggregate metric to precision, the most pressing questions are still not being addressed: is the company saving money by preventing fraud? Are the customers happy or annoyed with the detection system in place? Do they trust the company's systems and understand why they're doing particular things?

Notice that true performance and trust are inseparable and that one of the most striking differences between a prototype and a production-level model is the amount of trust one can confidently deposit in it.

Fortunately, there is a path towards performant and explainable machine learning. Shedding light into these black-box models start with proper error analysis.

*Aggregate metrics summarize the performance of a model across a whole dataset into a single number. They are attractive to practitioners because they are easy to interpret and convey useful information, but relying solely on them can be very problematic. The most popular examples of aggregate metrics are accuracy, precision, recall, and F1.

Chapter 2

Systematic error analysis

In the previous chapter, we have argued that trust in ML models is of utmost importance if we wish them to live up to their full potential. We are amid an era when there are high stakes and high expectations involved, but models are still seen as black-boxes. Furthermore, ML practitioners won't be able to avoid the mistakes their models make unless they start looking beyond aggregate metrics, such as accuracy or precision.

The path towards trustworthy ML models starts with treating **error analysis as a central component** of the development process.

> Error analysis is the attempt to analyze **when, how, and why models fail**. It embraces the process of isolating, observing, and diagnosing erroneous ML predictions, thereby **helping understand pockets of high and low performance of the model**.

Error analysis should be **seen as a systematic process that encompasses various activities**. In this chapter, we explore some of them, motivating their necessity and providing guidelines that shall help practitioners incorporate them into their ML development pipelines.

Error cohort analysis

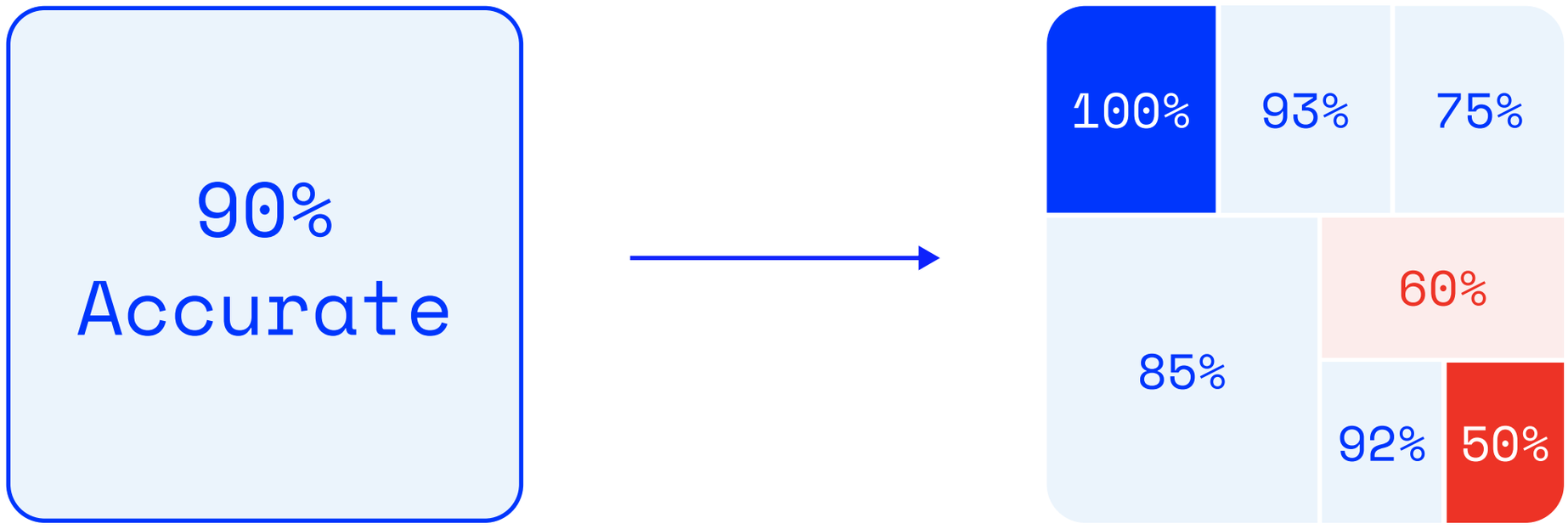
Imagine you have trained a classifier that predicts whether a user will churn or continue using your platform based on a set of features, such as age, gender, ethnicity, geography, and others. Now, you want to know your model's performance, so you assess its accuracy on a validation set. The accuracy you obtain is equal to 90%. That's great and you are feeling proud of your work!

The 90% accuracy, as an aggregate metric, summarizes the performance of your model across your whole validation set. It is a **useful first metric to look at**, but it certainly **doesn't convey the complete story** of how your model behaves.

Is that accuracy sustained across different subgroups of the data? For example, how does your model perform for users aged between 25-35? What about for users based outside the US?

Notice that from a business perspective, the answers to these questions might be very relevant, so you need to be confident that your model is coherent enough to answer them.

What you will most likely find out is that **the accuracy of your model is not uniform** across different cohorts of the data. Furthermore, you may even encounter some **data pockets with low accuracies and specific failure modes**.



If you looked only at the aggregate metric (the accuracy, in this case), you would have a myopic view of your model's performance and think that it was satisfactory. **This is why analyzing different cohorts of the data is critical to building trust in your model and not being surprised by failure modes only after your model is serviced in production.**

Global and local explanations

What has a model actually learned?
By looking solely at aggregate metrics, it is not possible to arrive at any answers.

> Being able to produce explainable predictions lies **at the center of trustworthy ML**. These explanations can be **global** or **local** and each one provides a distinct perspective to practitioners and businesses

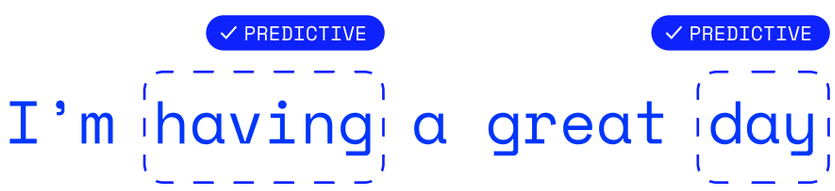
Global explanations **help reveal which features contributed the most** to the (mis)predictions made by the model over a dataset. In the churn classifier from the previous section, for example, one might find out that the users' geography is one of the most important features to predict whether a user will churn or not. Note that this information can be **directly translated into business insights**. A marketing team, for instance, might decide to create specific campaigns targetting the users from a particular location. It also increases the ML practitioners' confidence that the **model is behaving properly, without over-indexing to certain features**.

Local explanations provide **insights into the individual predictions** and help practitioners get to the **root cause** of problematic predictions their models are making. The justifications provided by local explanations build confidence that the **model is taking into consideration reasonable data to make its predictions**.

For example, in a sentiment analysis task in natural language processing (NLP), where a phrase is categorized as positive or negative, if a model predicts that the sentence "I'm having a great day" is positive because of the word "having" and "day" and not because of the word "great", is it really working?

Not really. Ideally, the model, in this case, should be predicting a positive label particularly because of the word "great", which is a strong positive word. Predicting the right label is only half of the story.

Wrong:



Right:

Explainability is an **active area of research**. Some models are naturally more explainable/ interpretable than others. For example, a linear regression is more explainable than a neural network, as in the former, the model's weights are evident and can be seen as proxies for feature importance, while in the latter, the high number of parameters and cascaded nonlinearities often make it hard to understand what's going on. There are other approaches that are applicable to a wider set of problems, such as LIME (Local Interpretable Model-agnostic Explanations) and SHAP (Shapley Additive Explanations).

As Marco Tulio, researcher at Microsoft Research and one of LIME's creators, puts it, "understanding the reasons behind predictions is quite important in assessing trust, which is fundamental **if one plans to take action based on a prediction**, or when choosing whether to deploy a new model".

When used in combination with a systematic error cohort analysis, explanations can be an important step towards understanding why the model might not be performing so well for a specific subset of the data.

Counterfactual and adversarial analysis

Will a model change its prediction if the values of a set of features are varied in an unforeseen way?

It probably will. The question that should be asked, then, is if such changes are **desirable or not**.

For example, in a model that assesses the credit risk of a user, the model's predictions should change depending on the user's income. But should it vary for different users' genders, all other features being equal? What about for distinct users' ethnicities?

It certainly shouldn't.

> Even though the answers are no-brainers, not all situations are as straightforward as that. **There are massive biases and failure modes hidden within industry-standard and academic-grade models and datasets**.

A good error analysis procedure should test the adversarial changes and try to find counterfactual examples where the model is not performing correctly. Without a proper counterfactual and adversarial analysis, these behaviors are hard to anticipate so that corrective actions can be taken in time.

For instance, a few years ago, a seminal paper was published indicating severe vulnerabilities of computer vision systems based on deep learning. It was shown that several systems were prone to adversarial attacks that were **imperceptible to humans, but could easily fool the models**. These results reinforce the importance of incorporating adversarial and counterfactual analysis to the error analysis procedures.

Synthetic data

One of the most challenging parts of building ML models is figuring out all the edge cases. Your training, validation, and test sets represent only a small fraction of the kinds of examples that your model will encounter out in the wild, after deployment. How can you prepare beforehand to deal with categories that are often underrepresented in the samples that you have?

Generating synthetic data to augment underrepresented portions of your training data is a great way to **increase your model's robustness, ensure important invariances, and further explore specific failure modes**.

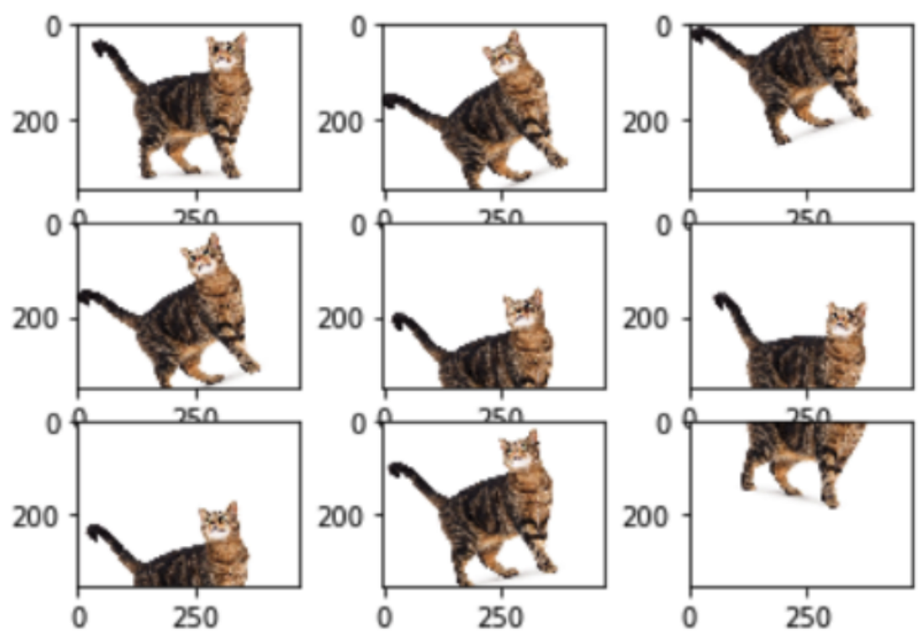
For example, in a sentiment analysis task in NLP, the phrases "John is a happy person" and "Mary is a happy person" should be classified equally. One can generate a large number of synthetic samples following the template "{name} is a happy person", where "{name}" is replaced by various common first names, to ensure the model is invariant to first names when performing the task.

There are various ways of generating synthetic data, depending on the datatype of interest. It can be **as simple as generating samples from a template**, as in the previous example, or **as complex as using generative adversarial networks (GANs)**.

Original data samples can also be perturbed to augment the dataset. In NLP, this can be done, for instance, by introducing small typos (which encourages model robustness to typos) or replacing word tokens with synonyms. In computer vision applications, this can be done by adding noise to sample images, changing image orientation, among a plethora of other ways. Each area has its own idiosyncratic methods of data perturbation for data augmentation.

The insights from the counterfactual and adversarial analysis can also be used to generate synthetic data, giving rise to **adversarial examples**. Being able to quickly come up with a set of adversarial examples either from an hypothesis or from modifications from the original dataset contributes to more robust and error-proof models.

Original data



Synthetic data

Systematic testing

Test-driven development is common practice in software engineering.
In ML, a field not that far away, tests are not as common as they should be.

Testing in ML (if done at all) is usually comprised of a single engineer writing a script to test a few cases that came up during a sloppy error analysis procedure. Thorough testing goes a long way in **ensuring model quality**, helping practitioners **catch mistakes proactively rather than retroactively.**

For example, borrowing insights from software engineering, Marco Tulio *et al.* to proposed the CheckList: a new testing methodology for NLP models. This work shows that "although measuring held-out accuracy has been the primary approach to evaluate generalization, it often overestimates the performance of NLP models". Moreover, "NLP practitioners with CheckList created twice as many tests, and **found almost three times as many bugs as users without it**".

Notice that each facet of error analysis presented so far can be systematized into unit and regression test frameworks.

Building over the ideas of error cohort analysis, one might define performance thresholds for certain subgroups of the data and test whether the model surpasses it for a dataset. From counterfactual and adversarial analysis, it is possible to build tests that strive to flip the predictions made by a model by manipulating the feature values. Finally, it is possible to create synthetic test samples that ensure the model's predictions remain invariant in particular scenarios.

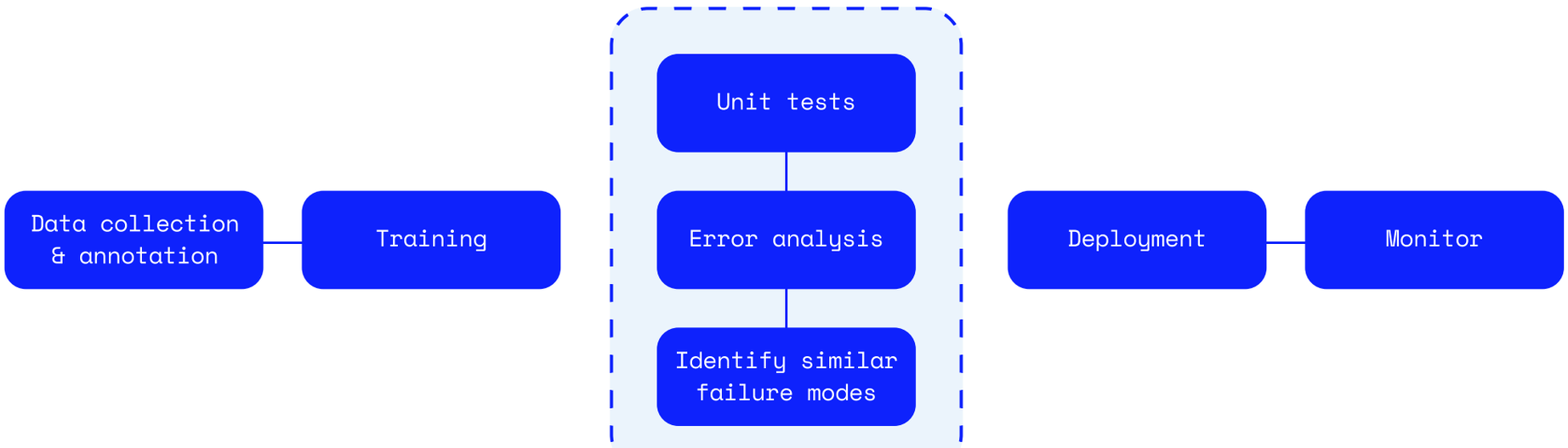
Chapter 3

Road to ML maturity: from L0 to L4

Working with ML models can be challenging: not only the problems are complex, coordinating the efforts of different teams around all the processes within data collection and annotation, model training, error analysis, deployment and monitoring **can waste a lot of resources if not done properly**.

Mature organizations understand that
systematic error analysis should lie at the center of all their efforts.

The insights that arise from it serve as a compass pointing in the right direction. As Stanford professor Andrew Ng puts it, “if you do error analysis well, it will tell you what’s the most efficient use of your time to improve performance”.



In practical terms, **how can one use error analysis to prioritize activities in an ML development pipeline?**

One of the key byproducts of a systematic error analysis is the **identification of a model’s failure modes**. As seen in the previous chapter, these failure modes arise naturally when performing error cohort analysis with (global and local) explanations and also after counterfactual and adversarial analysis.

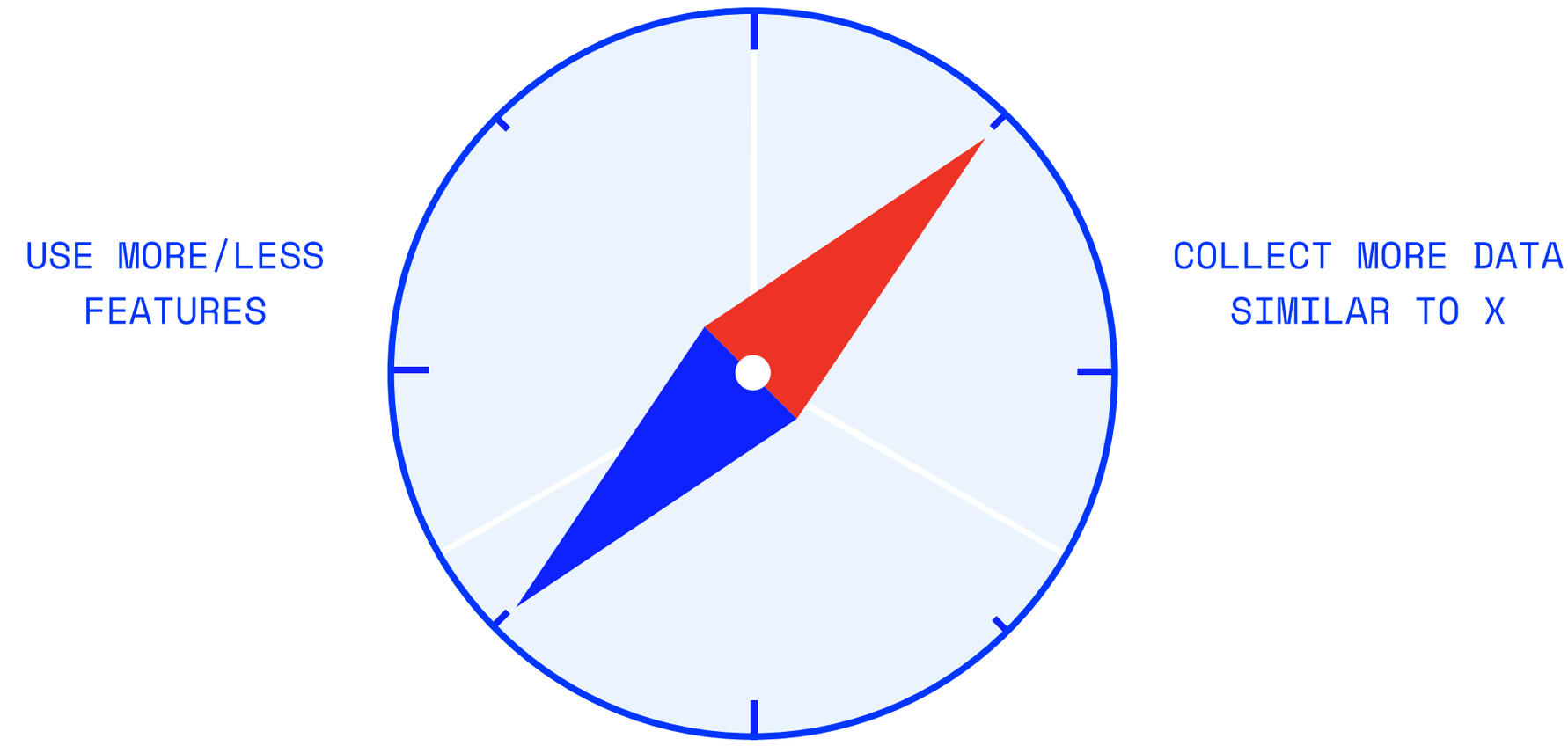
Now, consider, for example, the re-training process, where there is a production-ready ML model, but the team is going through an iteration of re-training with new data to improve the performance.

Using the identified failure modes to guide the efforts of data collection and labeling can greatly save resources. Being able to pinpoint exactly what kind of data is required to boost the model’s performance is a competitive advantage in an age where most organizations collect data in a half-haphazard way.

The insights generated from error analysis also benefit the other end of the ML development pipeline: monitoring. Instead of **spending profusely on monitoring solutions** and dealing with the never-ending anxiety of waiting for the moment a model will break in production, organizations that understand the importance of error analysis and systematic testing **deploy models with confidence**. They have a deep clarity of the failures and sucesses of their model. In short, they know what they are doing: building trustworthy ML.

Maturing doesn’t happen overnight. We have categorized organizations into five different layers, called L0 to L4, inspired by this blog post. Hopefully, this construction can help you assess **where your organization is** in terms of maturity dealing with ML models and point you to possible **next steps**.

The Systematic Error Analysis Compass



L0

Most organizations are still in L0. They train their models on a training set, perform model selection on a validation set and assess their models on a test set **using aggregate metrics**, such as accuracy, precision, recall, and F1.

If you read this far, you know how easy it is to be **misguided by aggregate metrics**. They often give you a false impression that you are delivering a high-quality model, when in fact, you cannot be so sure.

L1

Organizations in L1 are one step further. Besides doing everything their counterparts in L0 do, they also **document error patterns**, so that they are mindful of their models' present limitations. They **perform tests**, to further explore the mistakes that happen most often and they **calculate F1 scores per class**, aligned with their business needs, gaining, thus, a better grasp of the model they have in hands. Finally, they're monitoring incoming data to check for drifts.

L2

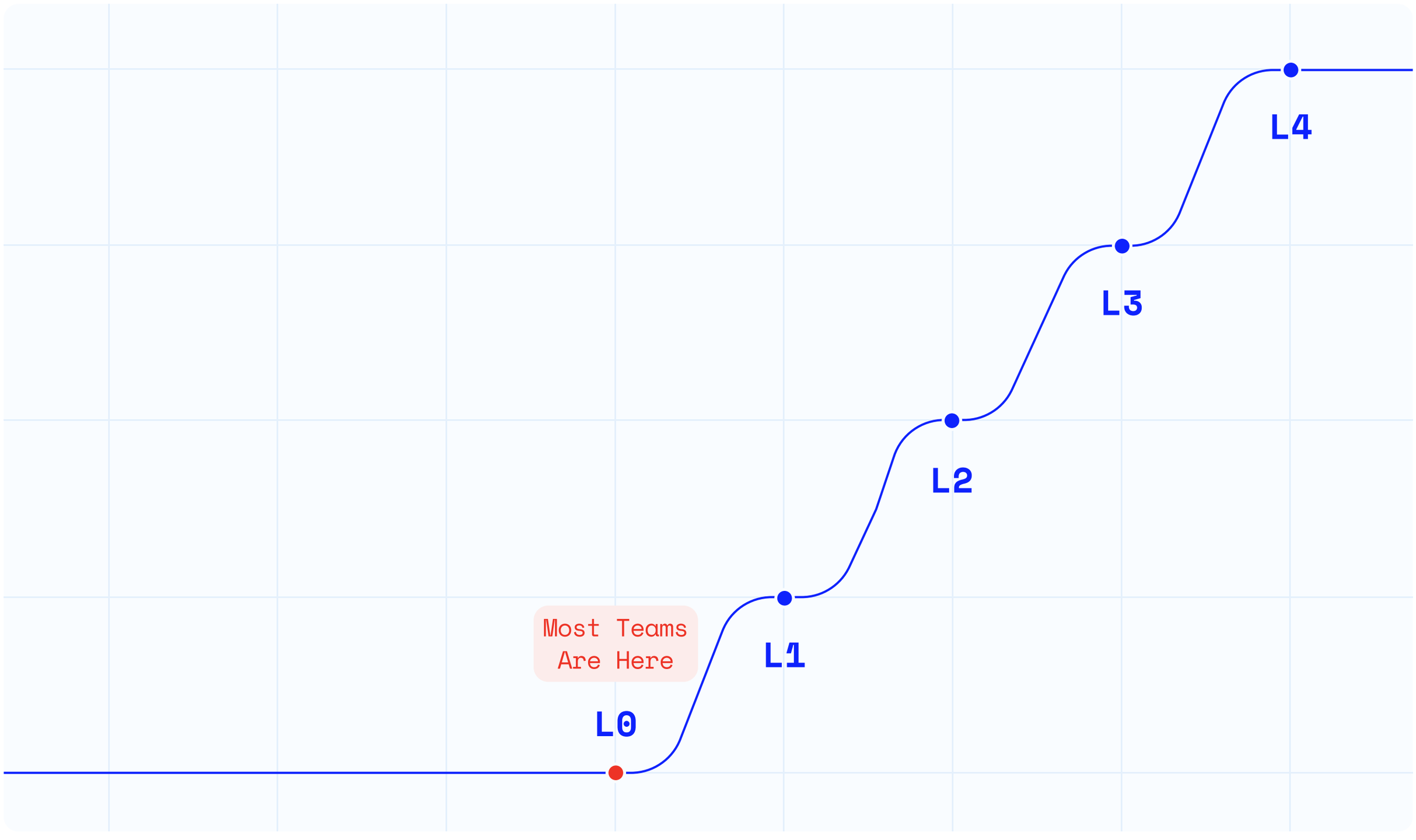
Organizations in L2 do everything organizations in L1 do. Additionally, they understand the importance of going beyond their original training data. They make use of **counterfactual examples** to challenge assumptions about the models. Furthermore, they rely on **global and local explanations** to check whether their models are indeed learning useful information and not paying too much attention to nonsensical characteristics of the features.

L3

In L3, organizations, besides doing the same as organizations in L2, **perform cohort-based inspections**, to ensure their model's performance is adequate across different subgroups of their data, and **what-if analysis**, to verify whether the hypothesis they constructed hold under particular scenarios.

L4

Finally, organizations in L4 represent the Holy Grail when it comes to best practices and building trustworthy ML models. Besides doing everything in L3, they are constantly **performing perturbation analysis**. They also **perform model diffs** and **data unit testing**, to ensure consistency.



	L0	L1	L2	L3	L4
Train/Val/Test split	✓	✓	✓	✓	✓
Overall Metrics, F1, P/R, Acc	✓	✓	✓	✓	✓
Document Error Patterns		✓	✓	✓	✓
Check most high conf. Mistakes		✓	✓	✓	✓
F1 per class aligned with business		✓	✓	✓	✓
Monitoring Incoming Data		✓	✓	✓	✓
Counterfactual Data			✓	✓	✓
Local and Global Explanations			✓	✓	✓
Cohort Based Inspections				✓	✓
What-If Analysis				✓	✓
Perturbation Analysis					✓
Model Diffs					✓
Data Unit Testing					✓

Chapter 4

Achieving trust with Unbox

We were once in your shoes. As ML practitioners, we have built ML models, deployed them in production and **have seen what happens when things go sour**. This is why we are working so hard to develop a set of tools that helps put error analysis at the heart of the ML development pipeline.

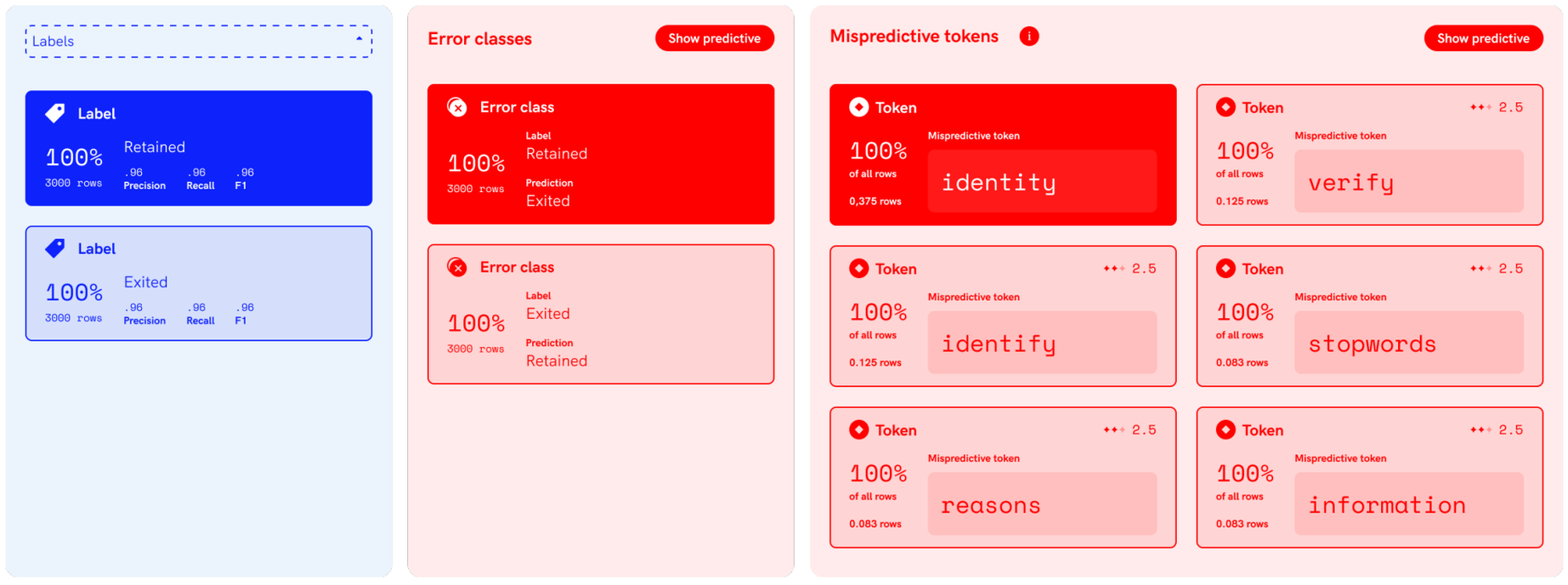
Unbox is the debugging workspace for machine learning,
where companies are able to **track and version models, uncover errors,
and make informed decisions on data collection and model re-training**.

By conducting a systematic error analysis, it is possible to start shedding light into black-box ML models. This process builds trust from every stakeholder: practitioners ship models with confidence, the business gains insights and serve happy customers, and the ML community as a whole thrives.

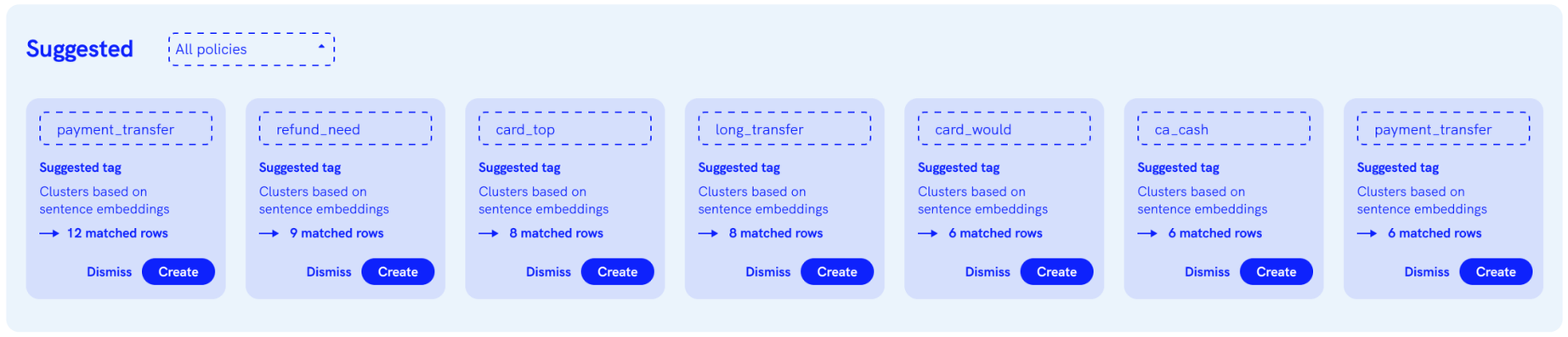
Slice and dice

As previously discussed, error cohort analysis **is critical to build trust in ML models**. The idea is that the model's performance is probably not uniform accross the different subgroups of the data and it is important to be able to unravel and understand the possible error pockets and failure modes **before** the model is shipped.

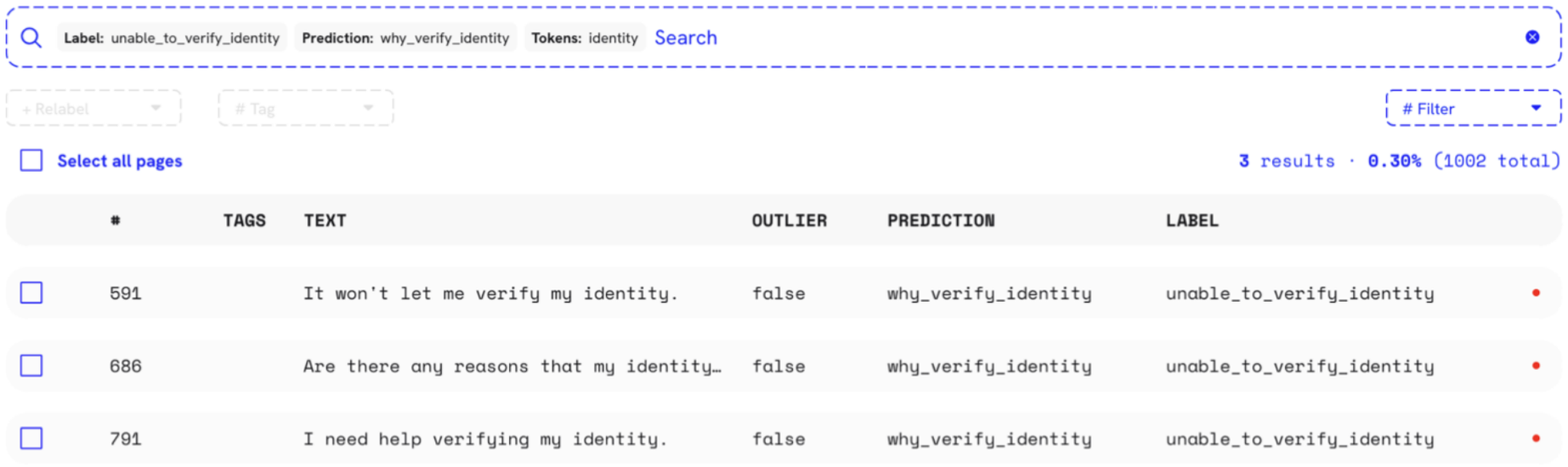
With Unbox, it is easy to perform exploratory analysis. You can start looking beyond aggregate metrics over the whole dataset and dive-into the error classes.



The **automatically suggested tags** give you a head start on the whole exploratory process. Combining flexible tagging with easy filtering results in endless possibilities to conduct **repeatable and precise data cohort analysis.**



Unbox

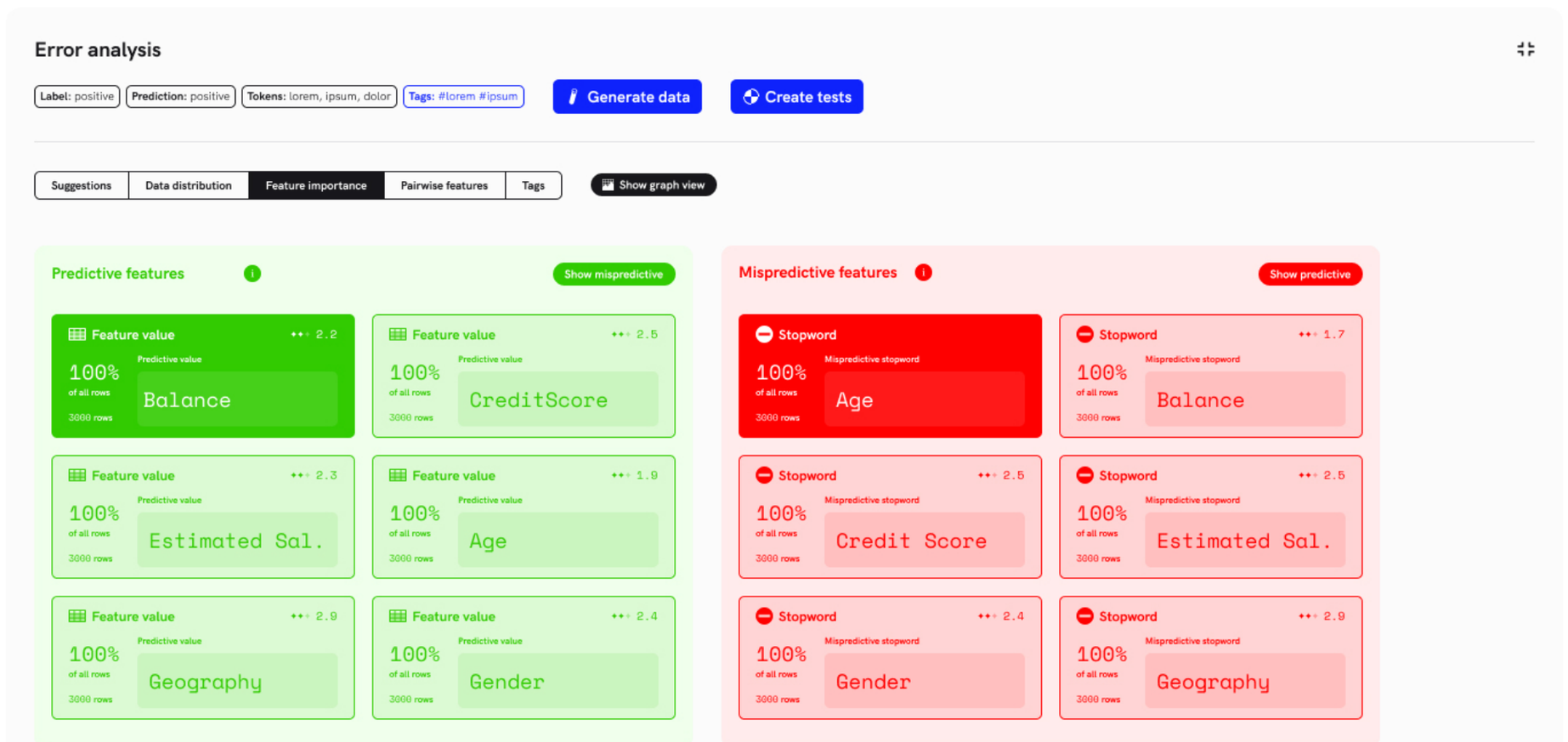


Built-in explainability

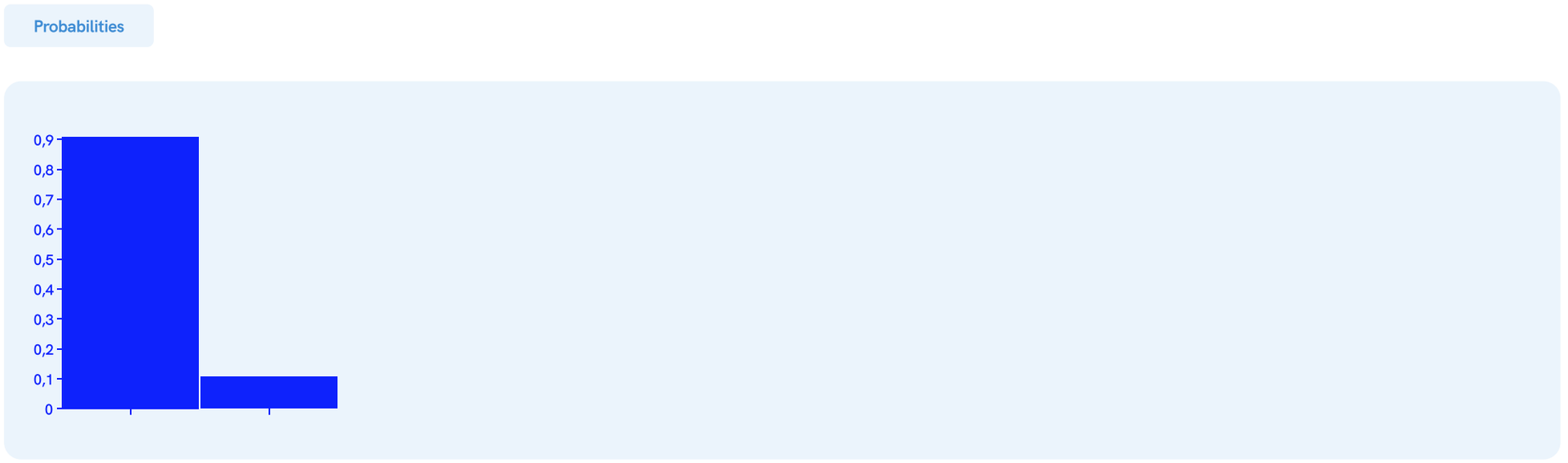
After uploading a dataset and a model, Unbox **automatically provides global and local explanations** to help you understand **why** your model behaves the way it does.

If you wish to **get to the root cause of a mistake your model is making**, you can take a look at the individual predictions and what motivated your model to behave that way.

You can get a glimpse into what are the most predictive and the most mispredictive features. **These are the features which influenced the most your model's predictions**.



If you wish to **get to the root cause of a mistake your model is making**, you can take a look at the individual predictions and what motivated your model to behave that way.



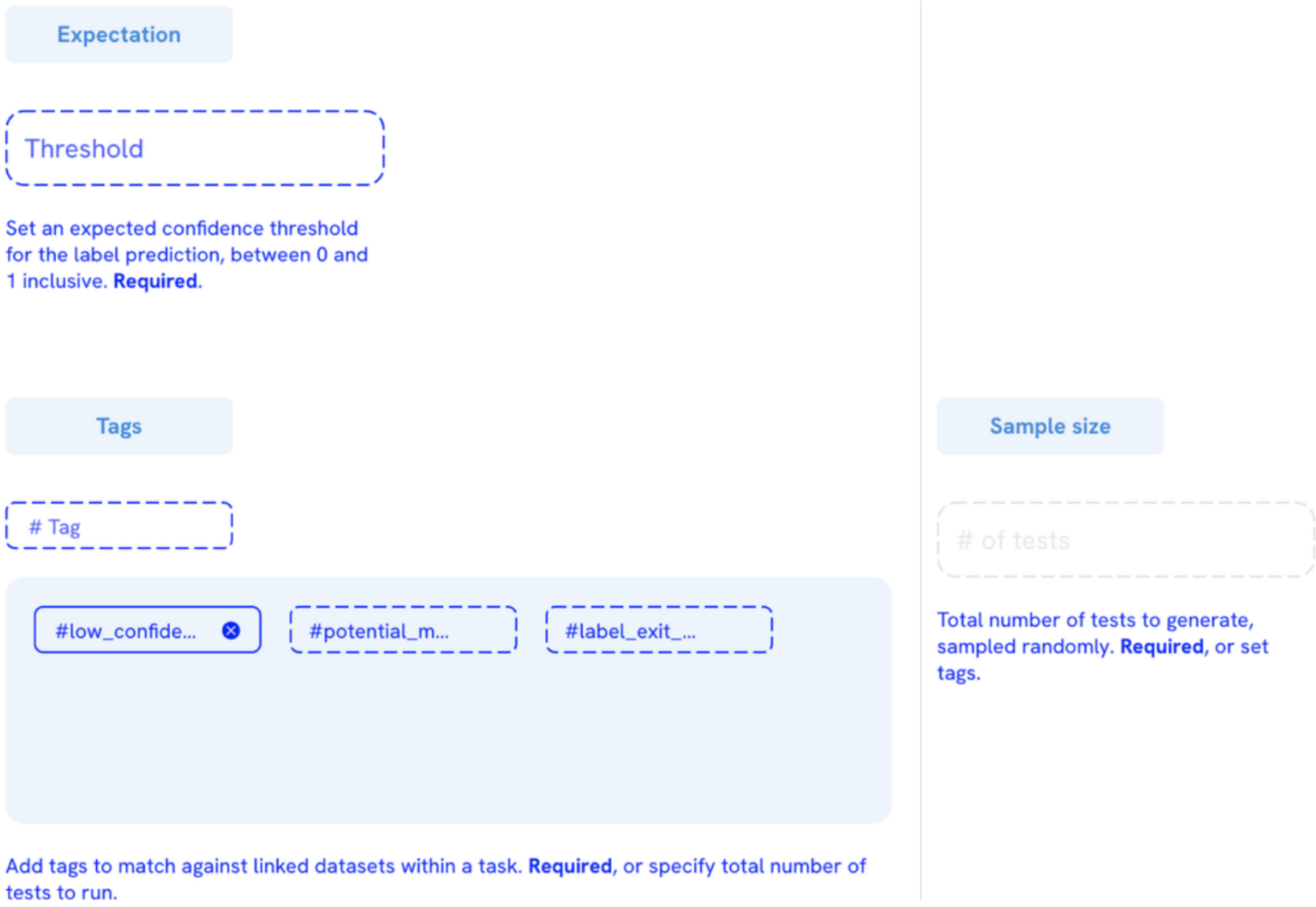
ML Testing

The only way to ship with confidence is through extensive testing procedures. With Unbox, it is possible to **create tests that evaluate your ML model in multiple ways**.

These tests are based on the procedures within error analysis, discussed in Chapter 2.

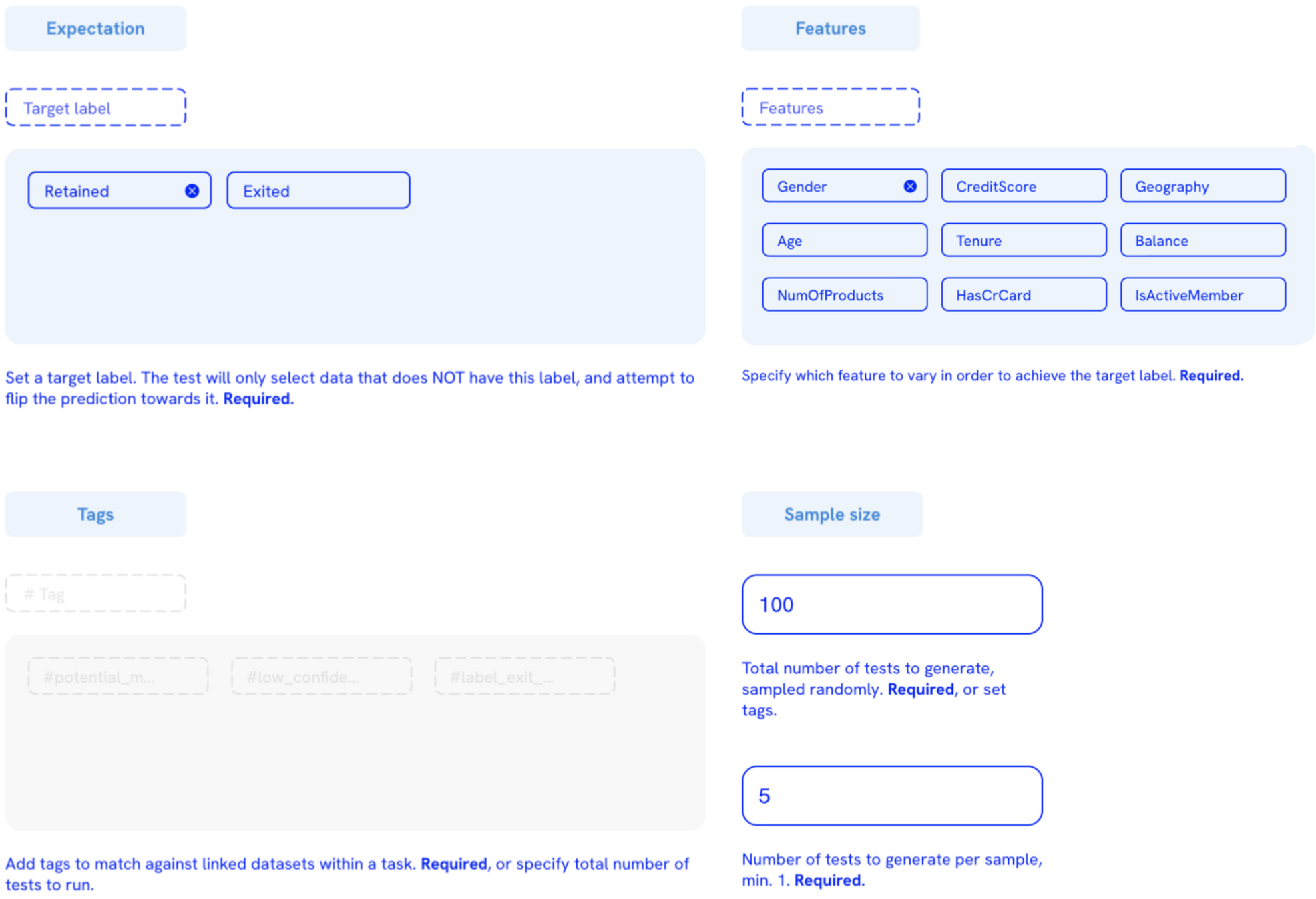
With **confidence tests**, which build over the ideas of error cohort analysis, one **can define performance thresholds for certain subgroups of the data and test whether the model surpasses it for a dataset**.On Unbox, you can easily create these tests by tagging the data samples that belong to the groups of interest and then defining the performance threshold you expect.

You can also choose not to test the model on tagged data samples. In this case, you can define a sample size, which is then drawn uniformly at random from your original dataset. This is also an opportunity to **discover** possible samples that might be **associated with problematic model performance**.

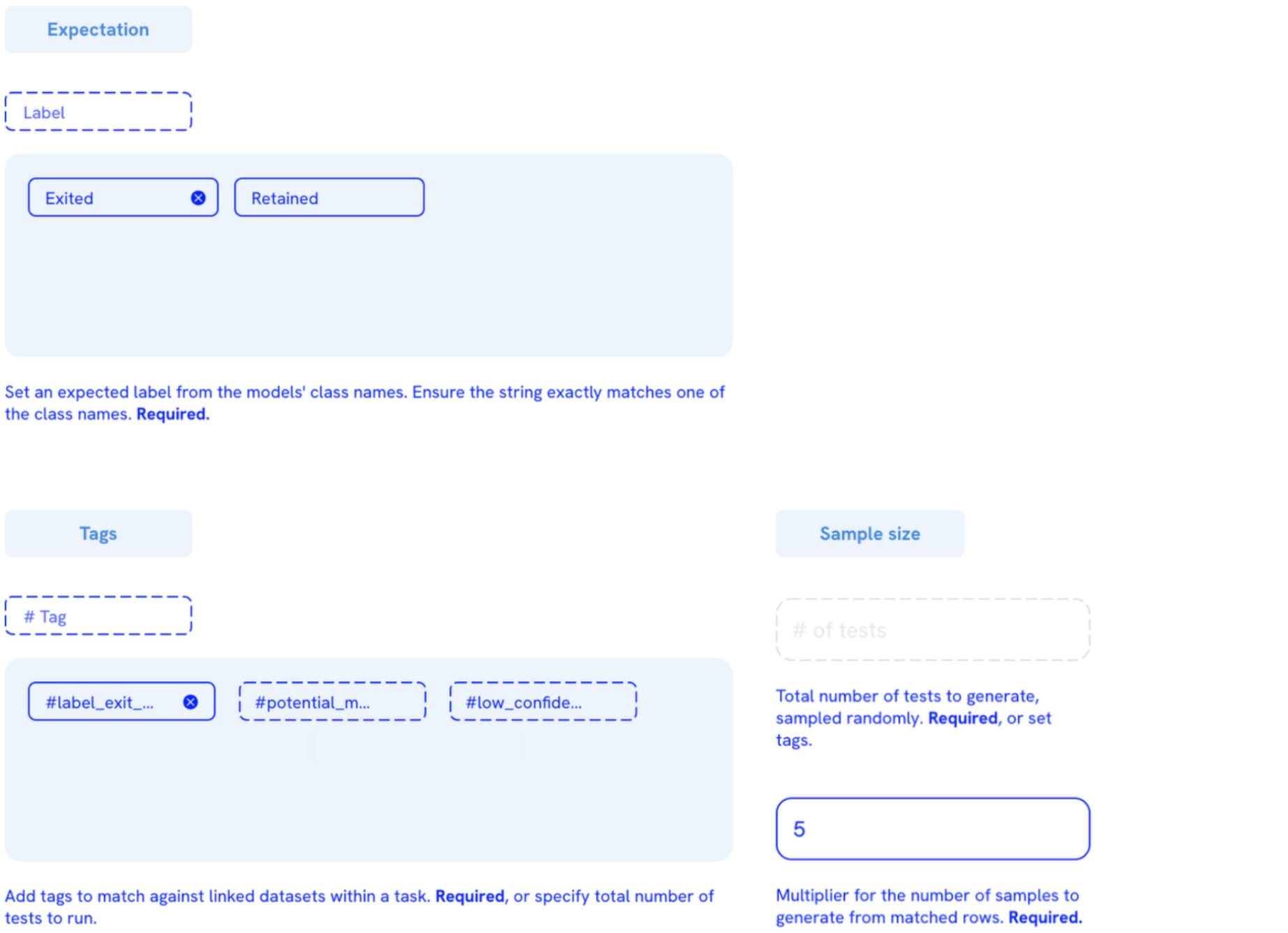


Building over the ideas from counterfactual and adversarial analysis, it is possible to create **adversarial tests** that strive to **flip the predictions made by a model** by manipulating the feature values.

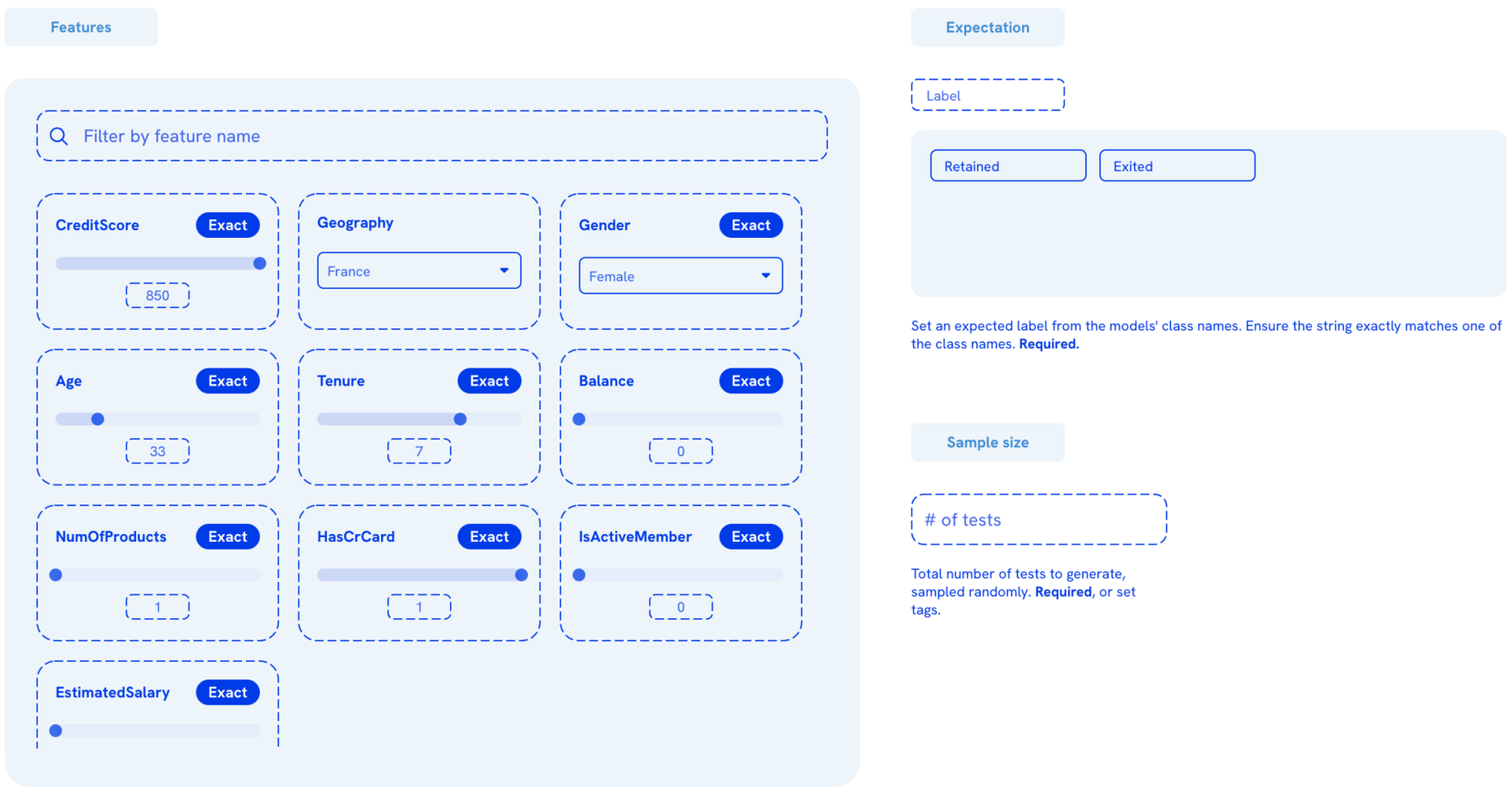
As a result, you end up knowing the feature threshold that needs to be surpassed in order to modify the prediction of your model to a specified label.



Data augmentation tests are a great way to ensure the model's predictions remain invariant in particular scenarios. With Unbox, the synthetic data is **generated from the existing samples** and the test verifies whether the **model's predictions are the same** as the one for the original data. The test can, as in the previous cases, be created from tagged samples or from randomly selected samples from the whole dataset.



With **unit tests**, it is possible to test whether the model **behaves as expected** for data that looks a certain way. For instance, with tabular data, one can select exact values or ranges of values for each feature and test whether the model's predictions are from the expected label.



Chapter 5

What can go wrong when you can’t trust a model

If you have read this far, hopefully, you are convinced of the importance of incorporating systematic error analysis procedures to shed light into black-box ML models. Moreover, we have mentioned (several times) that everyone is at loss as a consequence of untrustworthy ML models.

But *how bad can it be?* you might ask.

In this chapter, we go through a few real-world examples that illustrate some of the consequences of deploying ML models which exhibit biases or have previously unknown failure modes.

> In the best cases, the results are a piece of **bad PR and upset users**. In the worst cases, it can run organizations out of business or permanently **impact people’s lives negatively**.

Google search results

Google currently relies on complex ML models to answer to the user’s queries. From time to time, some of the search engine’s odd results end up gathering media attention. Recently, for example, there was a considerable controversy related to the image search results that come up after searching for “school boy” and “school girl”.

On the one hand, the image results for “school boy” show young boys dressed for school, as one would expect. On the other, the results for “school girl” show sexualized images of women wearing school uniforms. **The difference between the two queries is a single word, the gender.**

Zillow’s iBuyer

Zillow had become the go-to place when it came to house valuation. The “Zestimate” was an estimate of the price of a house produced by a model from a set of features provided by the user. After a while, Zillow created its iBuyer arm, and **automatically bought houses based on the Zestimate**.

The idea worked for some time and flipping houses relying on the Zestimate seemed to be a profitable business. Eventually, though, the scenario changed and things stopped working.

In 2021, Zillow **reported lossing over $500,000,000** as a result of buying houses **overpriced by their model** and selling them at loss. As a consequence, it shut it’s iBuyer arm and said it would lay off 25% of its workforce.

No one knows for sure who is at fault in Zillow’s iBuyer case. One thing is for sure: ML models are deployed on dynamic environments and simply monitoring them might not be enough to guarantee that they are working properly.

Predicting recidivism

Since the late 90s and early 2000’s, there are ML models being used to predict recidivism, i.e., models that strive to compute the likelihood of an offender commiting another crime. The results produced by these models are often **used to support judicial decisions**.

This is clearly a high-stakes situation: someone's life might be on the line, one way or the other, as a result of actions taken based on the output of an ML model. **But are these models trustworthy?**

In a famous case involving the risk assessment software COMPAS, it was discussed whether the model predicted a **much higher recidivism rate for black defendants** than for white defendants. In terms of accuracy and error ratres, the model indeed behaved differently for **distinct ethnicities**.

Since it was first developed in 1998, the model from COMPAS was used to evaluate more than **1 million offenders**. Whether ML models are appropriate or not for the task at hands is a discussion out of the scope of this paper. What is unquestionable is the impact that such a model had in the lives of so many people.

Chapter 6

Conclusion

As John Naisbitt puts it, we live in an age when “we are drowning in information, but starving for knowledge”. We have massive volumes of data being produced at every instant and **ML can serve as a tool that helps us make sense and leverage all that information**. If only bridging that gap was so simple...

Working with ML models is challenging and the whole process involves a cascade of decisions with consequences that are not completely evident at first. **To add to that intrinsic difficulty, ML practitioners are shipping black-box models everyday, without conducting systematic error analysis**. The consequences were laid out throughout this paper.

We, as practitioners, need to change. ML models are much more than their performance measured by aggregate metrics. **Error analysis should be at the heart of the ML development process and guide every effort that surrounds it**.

Stanford professor, Andrew Ng, recently launched the data-centric AI campaign. He argues that the process often followed by the industry **must be transformed**. Currently, the **data is usually held fixed** and the **model is improved iteratively** until the desired results are achieved. According to him, a more effective approach to get to the right results is the opposite: **holding the model fixed and iteratively improving the data quality**.

As a consequence, the lifecycle of data-centric AI consists on training the model, **conducting error analysis** to identify the type of data the algorithm does poorly on, either **getting more of that data** via data augmentation or additional data collection or giving a more consistent definition for the data labels if they were found to be ambiguous.

Andrej Karparthy, Tesla AI director, introduced the concept of “operation vacation”. The idea is that through a **systematic error analysis process**, it is possible to identify the samples from the long tail and the mistakes in order to re-train the model.

One of his famous essays talks about the concept of “**becoming one with the data**”. In his own words: “the first step to training a model is to not touch any code at all and instead begin by thoroughly inspecting your data. This step is critical. I like to spend a copious amount of time (measured in units of hours) scanning through thousands of examples, understanding their distribution, and looking for patterns. Luckily, your brain is pretty good at this. One time I discovered that the data contained duplicate examples. Another time I found corrupted images/labels. I look for data imbalances and biases.”

The path towards performant and explainable machine learning starts with proper error analysis. **If you are serious about it, you know where to find us.**

Unbox