firefly.ai

TURN YOUR PRODUCT INTO AN AI MACHINE

With These 3 Simple Steps

By **Erez Shilon**, Head of Product Management at Firefly.ai

TABLE OF CONTENTS

Step One

HOW TO CHOOSE THE RIGHT AI MVP

“WITH GREAT POWER COMES GREAT RESPONSIBILITY”

and for product executives that is no exception. Product leaders are constantly on the alert for ways to enrich the company product, and to add elements to strengthen its impact and increase the company’s revenue.

You’re aware of the huge buzz around AI and ML. You’ve heard that machine learning could significantly improve your technology through endless use cases such as fraud detection, demand forecasting, pricing optimization, customer behavior optimization, product personalization, and product recommendation, but with so many options, how do you choose the right use case for machine learning within your unique product?

It’s likely that some of the items in that list of commonly-adopted machine learning uses for technology products piqued your imagination, conjuring up visions of your tech on steroids, not to mention a few more zeros at the end of your quarterly revenue projections.

But it’s critical to choose the right use case for your MVP. The success of integrating machine learning with your product depends greatly on identifying where it can make the most impact and add the most value in the shortest amount of time.

Here's how it works:

A step-by-step guide for determining the right use case of machine learning for your product

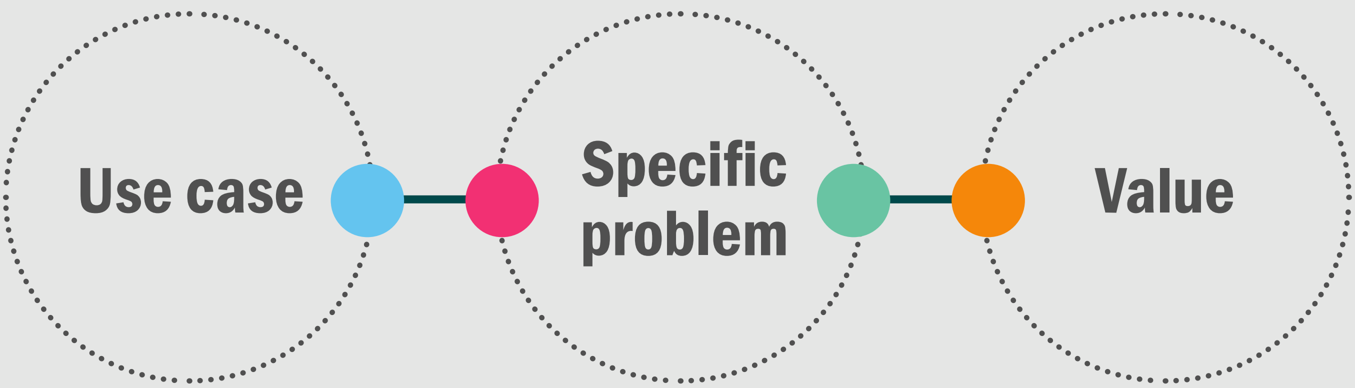
Step 1
What seems to be the problem?

First, consider the most pressing challenge around your product. Think about who might be suffering most from this issue, and how solving it would impact their work and wellbeing.

Let's take "churn", for example. Is this a problem that is impacting your business and your team? What value would a solution to this problem bring to your team, and to the business as a whole?

At one of my first product management roles, my team was building a trading platform and we noticed that we had concerning churn ratios at one of the onboarding steps (this is easy to find using analytics). However, understanding who will churn isn't that easy. You can try grouping users according to similar traits and build a rule-based system to detect who will churn, but this takes constant tuning and is never fully accurate. If we'd had a strong predictive analytics tool back then, we would have saved months of work, and the results would likely be more accurate.

Use Case Evaluation Process



Step 2
Do you have the right data?

At times you might have a fantastic use case in mind while lacking the right data.
For example, you wouldn't try to solve a churn or LTV problem without product or service usage data. If you don't know how a user is using or not using your product, you'll have a hard time building the right dataset for churn prediction.

What will be the ROI for your AI MVP?

In order to determine the ROI for adding machine learning to your product, you must perform both an impact analysis and a value estimation. This is the point in the use case consideration process where you link the machine learning initiative to your product KPIs.

This step is very important. If done correctly, it will enable you to convince other organizational stakeholders to help you with the productization. Who wouldn't want to take part in a new AI initiative that's easy to understand? Everyone would! They just need to understand the value that it will provide.

1 Perform an impact analysis

This is my sweet spot; it's what I do all day, every day, at Firefly.ai. Regardless of whether your product is business-facing or consumer-facing, it represents one or more user business flows. For this step, start by choosing the user business flow that has the most users.

Let's assume that you're a B2C product manager, and you're concerned with your conversion funnel in your product or application. If your use case is at the top of the funnel, it's going to impact more users. The deeper you go within your funnel, the fewer users you'll impact.

On the other hand, if you're a B2B product manager, you'll start with the areas in the product or platform that your users use the most, or those at the top of their business flows.

In fact, this process should be used for every product feature that you promote. So if it sounds familiar, you must already be implementing it, which is great!

2 Estimate the added business value

Once you've completed your impact analysis, you should think about the value. For example: If I present a "special offer" to a user with high LTV prediction, or a user who's about to churn, what value can I anticipate for the user?

Value can be measured in actual dollars (e.g. User LTV is $23), or in other metrics such as customer stickiness (e.g. It will be X times harder for the user to abandon my product). True, this is a high-level estimation (some would call it, "guess-timation"), but remember, our goal is to be able to compare the different options without getting caught in "analysis paralysis".

3 High level complexity estimation

Assuming that you've got the right data, understanding the complexity of using that data is very important for your initial success. Your first artificial intelligence initiative should be successful, so you can then gather momentum. Here are the important things to check for:

3A The ability to pull the initial data. Without this there can be no MVP.

3B The ability to continuously feed new data into the model. Without this there can be no long-term deployment.

Using these factors, you can start understanding the ROI for the AI MVP.

Next, compare the impact and value to the complexity of building an AI MVP. Once you have the basis for your decision making, it's time to compare a few problems, each with its corresponding complexity versus impact analysis and its value analysis. This is what you will be using to recommend business decisions to your stakeholders.

An example of an ROI comparison table:

USE CASE	DATA COMPLEXITY (1-5, 5 = EASY)	IMPACT	VALUE	OVERALL
A	2	3	1	2
B	4	4	3	3.66
C	5	1	2	2.66

Experiment time!

After you have identified the use case that will yield the highest impact at the highest value, it's time to plan the MVP. At first, you should regard planning your MVP as an experiment. A PM should approach an AI MVP initiative in the same way as they would build an A/B test plan.

How does planning your AI MVP initiative compare with planning your A/B tests?

When you plan an A/B test to prove a specific hypothesis, you wouldn't settle for only one test, right? (That could have some serious ramifications!)

With A/B testing, you'd typically plan a series of tests which would lead you to the winning option. The AI MVP isn't so different. When you have a hypothesis and need to test a few options, testing variations can range from the data itself to data science related "tricks" (e.g. feature engineering). This is where a data scientist can really add value to your process, because they've spent their entire careers working with data and ML models.

Success criteria

Specific metrics are used to measure ML models. These can vary from MAE and RMSE for regression problems, to Recall Macro and Precision for classification, for instance. Your data scientist knows this like the back of their hand. But as the PM, it's important for you to use your domain expertise to help shape the target metric to represent the value to the user as much as possible.

It is crucial to use the right metric for each model. Let's consider the previous churn prediction example. And let's assume that sending an offer to a client who's not going to churn is very costly (i.e. a retention offer might cause an otherwise satisfied client to churn, resulting in business loss). In this scenario, using the right target metric to reduce false positives will be critical to the success of your initiative. This is a great way for a product manager to align the company's business goals with the AI-initiative KPIs.

Experiment list

Make sure to plan your experiments from beginning to end, and leave room for tweaks should something new come to light. Establish a clear timeline, and identify specific success criteria. This is definitely something you'll want to include in your presentation to stakeholders.

How can Firefly.ai's AutoML platform help?

Personally, I love to test new ideas, but like most product managers, I can't do this all day. The beauty of AutoML is that you don't need to think so much about the model's development time. Once you have the right dataset and you've defined your use case accurately (like the awesome product manager that you are!), you could run as many experiments as you want on your way to building the right model for your use case.

A process that could have taken months (depending on the number of experiments) can now take mere days. You won't need to compromise on the model's accuracy by saying, "It's just an MVP; it'll be better once we go live".

Think of it as a new, refreshing, faster way to test your assumptions and embed machine learning capabilities into your product.

THERE'S MORE TO COME!

If you learned new things from Step One, you'll definitely want to keep reading Step Two and Step Three of this ebook, where we'll delve into how to deliver an AI MVP, and how to implement AI productization fast.

Step Two

HOW TO CALCULATE ROI FOR YOUR MVP

As a product leader, you're eager to find ways to improve your company's product offering. Now that you've identified the right use case, and carried out the value/impact analysis to prove that this is the best issue to focus on at the moment, it's time to move on to the next step.

Build an AI model and validate your MVP

This might sound daunting, but don't worry. Think of your ML process like any other product-validation experiment. Just like any other experiment, you need to plan it carefully, and be patient when things go wrong. Here's a quick overview of the steps to validating your MVP:

1. Define the problem (no worries here, you already completed this step), and gather your data
2. Build an ML model that matches your needs
3. Measure the model to check its accuracy and prove that it's an appropriate choice for your use case
4. Test the model on real-life data
5. Prove the value of your model by demonstrating real-time KPI uplift

Now let's discuss this process in more detail.

Step #1 Define the problem and gather the data

You've already defined your problem and delved deeply into it, so you can skip straight to the second part of this step, which is gathering the right data to solve the problem. This is possibly the most critical part of the entire process, so don't rush. Your AI model will only be as effective as the data you feed it, so don't supply it with junk.

When it comes to your MVP, feel free to get your hands dirty, but try to keep your data clean.

As the domain expert, you're strategically placed to draw on valuable data. You can get creative with feature engineering and pull data from as many different sources as you'd like, as long as you follow the golden rule: keep the data clean.

Being the professional that you are, you probably already know to avoid rookie mistakes like using illegal features, such as "number of clicks" on a landing page button as a feature for predicting landing page views (since a click is the result of a page view, we cannot use the number of clicks to predict page views).

Step #2 Build the model

Now we come to the tricky part. Take a deep breath, and turn to the many tools and open source libraries at your disposal. Resources like SKLearn and TensorFlow are here to help you build your own machine learning model without tearing your hair out.
(By the way, it's a good idea to memorize these names, because you'll encounter them repeatedly. Knowing them makes you look good in front of your data scientists!) Those who are comfortable with coding may want to use tools like Amazon Sage Maker or Azure Machine Learning Studio.

Either way, consider this a great opportunity to educate yourself on the basics of machine learning. Read up on how to prepare your data for ML models, how to build a data pipeline, and different algorithms and how to calibrate them using hyperparameters optimization. For this part, you'll want to keep your data scientists near to hand.

Step #3 Measure your model

Once you've built the model, it's time to measure it. There are a number of different metrics that you can use, so the challenge is to find the right one that fits your business use case. Consult the internet to find explanations and techniques for measuring your model.

Whatever metrics you utilize, the most important thing is to split your data into separate datasets. You could split it into two datasets, one for training the model and one for testing it, or into three datasets, for training, validation, and testing. Take care that you distribute your data correctly between datasets, to avoid leakage or overfitting.

Step #4 Test your model with real life-data

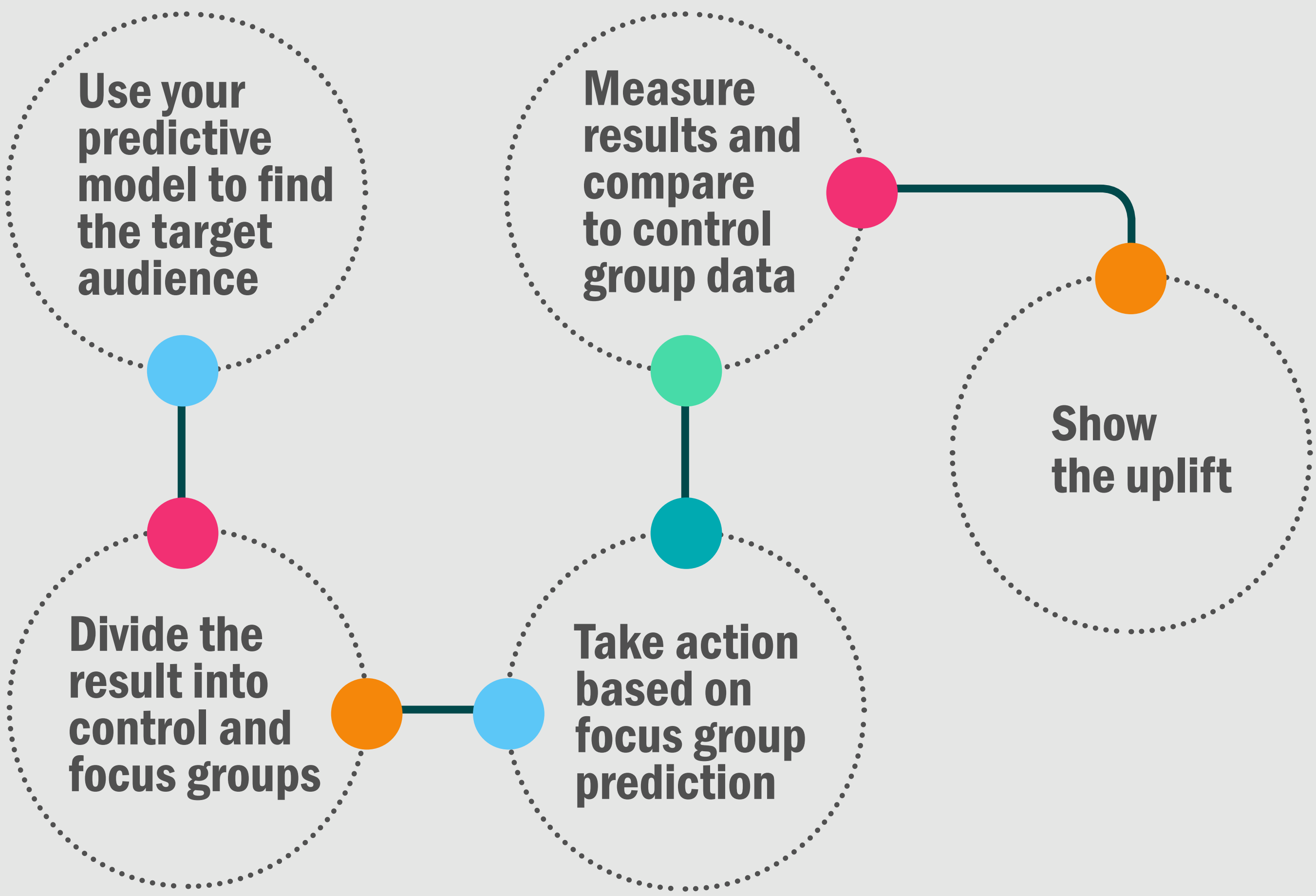
Measuring your model with test data is a vital step, but the best way to really check its performance is by running it with data drawn from real life. You can take some sample production data for which you already know the target result, and run them through your model. This will show how accurate your model really is, and help you to prove the value uplift of your MVP.

It's also a good way to build your confidence about your MVP, which is important when it comes to presenting it to your stakeholders to convince them to buy in. What's more, once you see that your model is working well on your production data, you'll feel a real sense of achievement. There'll be no stopping you!

Step #5 Prove uplift

If you pull your test dataset from production data, you will see how your model's predictions affect your bottom line. For example, in one of my previous product management positions, we built a job application process. To increase the ratio of applicants per job listing (one of the company's main KPIs), we tested this hypothesis: By adding an "apply to multiple, similar jobs at once" option at the end of one job application, the company would receive more job applications for each job they were hiring for. To that end, we built a model which aggregated similar jobs. Before deploying it to the entire job base, we divided the open positions into two groups, just like A/B testing. We set the control group aside and showed similar jobs only to the focus group.

After two weeks, we compared the new application rate with the previous application rate and the findings showed a significant uplift of applicants per job, while still maintaining the validity of the applications (i.e. someone who applied to a product manager position, would only be offered other product manager positions to apply for). This is one way of tying your machine learning model results to your business KPIs, and it's a great way of demonstrating the value to your stakeholders.



How can Firefly.ai's AutoML help?

As you can see, building an ML model isn't easy. Possibly the hardest part is finding the best model for your data. Master data scientists develop a level of intuition for selecting the best model, but even they have to spend time working through the process of trial and error until they reach the best option.

That's where AutoML comes in. AutoML tests different algorithms with various hyperparameter combinations to find the best solution. AutoML even helps you to prepare the data and build the right pipeline to train your model. At Firefly.ai we also invest heavily in optimizing the model search until it finds the ideal model for your needs.

Firefly.ai enables faster testing, since you don't waste as much time building the models, and helps you to validate or disprove your assumptions rapidly. For example, if you build a model and don't get good results, how will you know if the source of the problem is the model hyperparameters or the data? Firefly.ai tests multiple combinations concurrently, helping you to find the best combination of data for each model.

THERE'S MORE TO COME!

If you found this article helpful, you'll want to keep turning the pages to Step Three of this ebook. Look out for our discussion on how to implement AI productization fast. You can also revert back to Step One by flipping back to page 3 of this ebook, in which we discuss how to spot the right use case for your product.

Step Name	Classic Modeling	AutoML
Define the problem	Define your problem, and gather clean and relevant data	
Build the model	Build a model that is relevant for your business problem, build a data pipeline, and fine-tune the hyperparameters	Will automatically test algorithms, select the most relevant model, build pipelines, and try different hyperparameters. No need for coding or machine learning expertise
Measure the model	Find the right metric to train and test your model's performance	Will run multiple tests simultaneously on multiple metrics for faster validation
Test your model	Draw on real-life production data to see how your model performs	Will test quickly using UI instead of coding and/or implementing APIs
Prove uplift	Run your model on new data to predict KPIs and deliver value	

Step Three

HOW TO TAKE YOUR AI INITIATIVE TO PRODUCTION

You know that feeling you have when you finish creating something? It can be something small like washing the dishes, or something larger like building a closet. We all have projects we're working on, and one of the coolest features we humans have is our ability to feel joy in our accomplishments. After choosing the right use case for your first AI initiative, you will have had that tingle. After launching your first AI MVP successfully, that tingle will have become a warm feeling. Well, get ready to party: You're about to take your first AI initiative to production. When that happens, bring a cake to your team—it's time to celebrate!

At its core, a machine learning model is like many of the products you use or manage; it has an input and an output.

In Step Three of this ebook, we're going to discuss how to take the wonderful MVP you built and bring it to production. You'll obviously need to convince some stakeholders and prioritize this effort amongst other roadmap items, but once you launch the MVP, measure it, and prove a KPI uplift, most of the heavy lifting will be behind you. Now—it's time to deliver.

Before laying out the steps of taking your machine learning model to production, let's discuss what a machine learning model actually is. It's more similar to the product you manage than you may think.

In our case, a machine learning model is basically a piece of code. The "input" is a sample for prediction: Each sample has the same features (data fields) you use for the model training, but without the target listed (because this will remain blank until the model provides the prediction). The "output" is the predicted value for the target. So with the "input + output" of a machine learning model in mind, consider that the primary requirement for a model to work in production is that the model provides a prediction (output) when you send a sample for prediction (input).

Send the sample data to your model

At this point, your goal should be to continuously bring new sample data to the model. There is a difference between gathering data for model training, which shouldn't be a frequent task, and sending new samples for prediction, which should be a continuous and stable process.

For example, when it comes to churn prediction, you would train your model on past data. When you have a new client, "input" the new sample data into the model. The "output" is the prediction whether or not a new client will churn. You should speak with your data architect about the best way to aggregate data and "deploy" it to the model you built.

How to add your model to your product: Coding

Working with open source libraries such as TensorFlow or Scikit-learn, once you have the model's code, you should add it to your production code base. Some open source libraries will provide model wrapping capabilities with API documentation, but either way, you'll need to write and maintain the code within your code base.

How to keep your model updated: Refitting and retraining

Before we move forward, a few words on model "refitting" and "retraining".

What is refitting?

To keep your machine learning model up-to-date, it's recommended to "refit" it with new data every once in a while. The refit action involves taking new data and sending it to the model in order to calibrate the model's hyperparameters. This way you keep using the same model, but gain new "settings". How often is "every once in a while"? It varies between use cases. If you're trying to predict the performance of online ads, you would refit your model every week. Some would even refit it every day, and I've heard of less than once-an-hour refitting. If you're trying to predict SaaS churn, typically once every 3-6 months is sufficient.

What is retraining?

Since refitting is more about changing the model's settings, it doesn't really challenge the status quo. When we want to really make sure that the model we built is up-to-date with the current data behavior and other trends, we retrain the model. Just like when we built the initial model, we would take any new data and start training the model. This is a longer and more complex process than refitting, and usually would be performed less frequently (every 6-12 months). Think about it as a sanity test that you should do to make sure your model is relevant.

As you can see, keeping your model up to date requires effort and maintenance. The day-to-day experience is very similar to software versioning management. When you have a software product in production, you work to improve it in your development environment, and once you have tested it and it's ready to go, you deploy it. The only difference is that instead of writing new lines of code, you either refit (change your model's settings) and deploy the new version to production, or you retrain (create a new model) and deploy it to production. Standard software management procedures like sandbox testing (testing a new software's functionality before it's deployed to production) should be applied here as well.

How can **Firefly.ai's AutoML** make a difference?

Before AutoML platforms entered the scene, refitting and retraining models had to be performed manually by a data scientist. Now that AutoML has joined the party, refitting is a breeze. Most AutoML platforms provide an API layer that allows users—all data professionals, not just data scientists—to refit and retrain models on an ongoing basis, which is provided by the platform UI (no coding needed).

An AutoML platform will provide its users with a solid API/SDK layer, which will allow them to send sample data (input) and get predictions (output) from a cloud-hosted model, or download it to a "docker" and deploy it locally. This really simplifies the process of code maintenance, especially if you'll need to refit or retrain your model often. At Firefly.ai we understand that refitting and retraining models is critical for prolonged success. We enable you to perform these asks via API/SDKs on an ongoing basis and automate the entire process.

I hope you found this ebook helpful. Personally, I'm a fan of keeping it simple. That's what I tried to do here: help fellow product managers make the leap and realize that introducing AI/ML to products isn't as hard as it's made out to be, especially if you have a tool like AutoML.

From one product manager to another, feedback is what keeps us going, right? I would really appreciate hearing from you, especially if you have any questions or comments about this ebook. And if you think I can help you introduce ML to your product, I'll be glad to do it.

Erez Shilon is the Head of Product Management at Firefly.ai. He likes solving problems that involve people and data. Previously, Erez managed both products and the people who manage them. He led the development of machine learning-driven products and data decision-driven teams. Erez holds a B.Sc in Industrial Engineering and Management from Ben Gurion University and an MA in Law from Bar Ilan University, and in his free time he enjoys free-diving and building things.

GLOSSARY OF TERMS USED

Term	Explanation
AI	AI stands for artificial intelligence. It refers to anything done by a machine that we consider requires intelligence. Generally, AI should include learning, reasoning, and self-correction to count as true AI. There are many types and approaches to AI—click here to learn more about them.
ML	ML stands for machine learning. It's one of the many types of AI. Machine learning programs use algorithms to learn from their own data and then predict future patterns without being specifically programmed. ML plays an important role in many apps. To learn more about ML, click here.
LTV	LTV stands for LifeTime Value. It's the way that SaaS and B2C companies measure the total net profit they can make from a single customer. Click here to learn more about LTV.
Churn	Churn, or churn rate, is a way of measuring how many customers leave your service each year. Businesses that provide subscription-based services use churn as part of their calculations for a customer's LTV. To learn more about churn, click here.
ROI	ROI stands for Return On Investment. It's a way of calculating the overall value of a tool, process, or investment to check whether or not it's worth the cost. Click here to learn more about ROI.
MAE	MAE stands for Mean Absolute Error. It's a way to measure the difference between two variables. In the world of data analysis and ML, MAE is a common way to calculate the gap between the prediction and the true value. Click here to learn more about MAE.
RMSE	RMSE stands for Root Mean Squared Error. Data scientists use RMSE to measure the accuracy of their data models, by calculating the difference between the predicted values and the actual values. It's often interchanged with MAE. To learn more about RMSE, click here.
Feature engineering	Feature engineering is a key step in applying ML. A "feature" is an attribute shared by all the data units used by the ML model. Finding the right features is key to creating an effective ML model, but it's also difficult and time-consuming. Learn more about feature engineering here.
SKLearn and TensorFlow	SKLearn—also called SciKit Learn—and TensorFlow are free, open source software libraries for ML and deep learning workflows, models, and algorithms. SKLearn and TensorFlow also work together as SciKit Flow, a simplified interface to help people get started on predictive analytics and ML. Learn more about SKLearn and TensorFlow here.
Hyperparameters optimization	Hyperparameters are the rules that govern training algorithms. The success of a training model—and thus the entire ML model—depends on optimizing the hyperparameters to the correct settings. Learn more about hyperparameter optimization here.
Data splitting	Before data analysts can use data for machine learning models, they need to split the data. Usually, data is split into three datasets: a training dataset, a validation dataset, and a test dataset. Learn more about datasets and data splitting here.
Leakage and overfitting	Overfitting is when an ML model produces training results that are too accurate, so much so that it can't produce reliable results with real world data. Leakage means that unexpected information leaked in to the training dataset, causing unrealistic results. Leakage has a number of different causes, and it is itself one of many causes of overfitting. Learn more about leakage and overfitting here and here.