# Machine learning project example – Building a model step-by-step

Machine learning is an intriguing field for a variety of reasons. Researchers believe that it will lead to the discovery of artificial general intelligence; it employs cutting-edge technology from a variety of fields, such as computer science, statistics, mathematics, among others; it's also the path to a very lucrative career.

Since you're here, we assume you're a computer programmer wondering how difficult it would be to break into field (whether you're a fresher or veteran), a statistician or mathematician curious about how much programming is involved in machine learning, or a management graduate or business analyst looking to gain a deep understanding of the field.

This is why we've put together a detailed view of the machine learning workflow. We'll do this by closely examining the decisions a machine learning professional needs to take every step of the way, using a fairly basic example. This should give you clarity on the gap between your abilities and what machine learning requires, and, therefore, help you assess the effort you would need to put into becoming a machine learning expert.

## The problem: Fraud detection at a bank

As a machine learning expert working for a large bank, you're tasked with building a machine learning model that can detect fraudulent transactions. You know you have access to a lot of data, from personal information to transactional information, and are aware that you could use it to build a solution.

## High level plan

You go about building a machine learning model the same way as you do any project.

A. You dwell on the problem, identify potential ways to solve it and set goals for yourself.
B. Once you know the direction the project must take, you involve the people who will be supplying you with the data.
C. The data is then cleaned for the solutions that you think will work.
D. You build a prototype of the model.
E. You test the results of the model. Most likely they'll be unsatisfactory, requiring you to go back to the ideation or data cleaning or prototyping stage.
F. After many iterations, you will have achieved the goal you set yourself in 'A', at which point the model is ready to be rolled out.

## 1. Setting goals for yourself

At the very beginning, it's good to set goals. After all, you know that the best solution will be found over time, not at the first attempt. The objective, therefore, is not to introduce an algorithm that will eliminate fraud (this will be almost impossible, anyway). Instead, it is to experiment with machine learning to productivity benefits to the fraud detection team. The accuracy and complexity of the algorithm will be boosted over time.

Your initial aim is, therefore, to build a model that can simply point out anomalous transactions to the fraud detection team. So if there is a netbanking sign-in at Delhi airport one morning, and a credit card transaction at a restaurant in Shimla that afternoon, it would need to be flagged off.
At a later stage, however, the algorithm would be refined to improve its accuracy. It might begin to track the IP addresses of all airports and assume that people that use their cards at those locations will likely make transactions in another city as will be taking a flight. Or it may be able to determine that a particular user making purchases at different cities is actually an airline pilot.

These refinements will happen over time. When you're starting out, though, your goals are simply to put in place a working solution that significantly increases the productivity and accuracy of the fraud detection team. This is important because, even though machine learning is an iterative process, it is essential to draw the line somewhere (or you'll forever be making improvements).

## 2. Gathering the right data

Banks collect a lot of data. As you're building the model for a bank, you would have thousands of data points to choose from. You, however, have some domain knowledge to narrow them down, or at least enough knowledge to engage in meaningful conversations with the product and business teams to understand how each data point could contribute to your model.

Machine learning experts should, therefore, have a good understanding of their domain in order to be effective, and be good at drawing insights from the rest of the organisation. Once you know which inputs you want to begin working with, you can begin with data preparation.

## 3. Data Preparation

Machine learning algorithms are completely data-dependent. Some perform better with small datasets, others may be better off with large datasets, but all of them require clean data. This requires transformations of the highest order.

While you will lead this task, it's likely that you will turn to a data analyst around this point. This is because the work involved in turning data into inputs that can actually be used in your model is significant. Let's delve deeper into what this may involve:

1. You will ensure that the use of various databases (email lists, customer account details, loan details, credit card account details) are consolidated, so that each customer appears only once on the list. Any duplication is to be avoided.

2. In case any of your data points is text, you will likely recommend that it be mapped it to numerical values before you can proceed, depending on the algorithm that is used. If you're using a lot of text, you may ask for the bag of words model to be used.
3. As you're also using images as a data point (say, CCTV shots in an ATM), you will ask for a smaller version of the image or a cropped version of it.
4. You will ask for raw data to be in alignment as far as possible, to make it easier to work with at a later stage. For example, the age of the customer in months and customer relationship in days may need to be transformed to years for simplicity.
5. As you are missing a certain dataset, you ask for it to be scraped while checking if you could buy sample data (however, you know that it isn't ideal to invest too much time or money in any such data point, as there's no clarity on whether it's valuable just yet).
6. At all costs, you will avoid invalid or inaccurate data. All records need to be fully updated to ensure that the predictive power of the model is not impacted.

As we're working on a supervised learning problem (fraud or legitimate transaction), you would need a sufficiently large and complete dataset to train the algorithm to differentiate one from the other. This is a lot of work and will take time. And we haven't even begun figuring out which algorithm will be the right one for our model.

## 4. Building the prototype

Once you have your data ready to go, you can think about actually working on the model. Beginners should note that this requires a lot of thought and patience, as finding the right features or algorithm isn't necessarily easy.

**Feature Selection:** For the problem at hand, we can have thousands of data points. Each of these data points is ultimately categorised as features. It's up to us which of these features will be most useful.

Of course, at a high level feature selection is done at the point of data collection. But, based on the algorithm you're using, and the number of features it can handle, there is an amount of feature selection to be done here, too. If you can identify a small set of features that can alone help you build a model, you will be able to work with a simpler model that performs really fast, is easy to maintain and to understand. So if your data is a simple representation of the business problem, the model is likely to also offer better results.

**Algorithm Selection:** Algorithms have their areas of expertise. So when considering which one to use, you would find that a small change in how you define the problem could drastically affect the set of algorithms you could use. For example, if you want the algorithm to classify transactions as fraudulent of legitimate, you would only use a supervised learning technique; however, if you wish to groups the lakhs of customers into 10 or so clusters based on the type of transactions they have, you would use a clustering algorithm, which is an unsupervised learning technique. The algorithm that works also depends heavily on the size of the dataset, number of features, whether you need the predictions to be in real-time and much else.

To get our model ready, we will need to play around with the features and test multiple algorithms. Given how complex the challenge of building a fraud detection model can be, you could be stuck at this step for a long, long time. You'll often hear that a particular team has been stuck at deciding which algorithm to use for weeks. But then again, you may find the right results in your first attempt.

## 5. Testing the model

Once you have the prototype of your model ready, with the right features and algorithm, and reasonably good results, you will assess the performance of the model, based on pre-defined metrics. It is at this stage that you will tune parameters to optimise the performance of the model (using a boosting algorithm, for example) .

It is also here that you will for the first time test the actual predictive performance of your model. On testing its performance, you will quickly know how far you are from your stated goal. You may realise that you need many more data points to actually build a model that brings a significant improvement to the fraud detection team. You may also think of some external data points that would improve performance in certain areas. Depending on the outcome of this test, you would go back to the ideation phase, data gathering or the prototyping phase.

Let's, however, spend some time discussing the metrics you would use. As the fraud detection problem is usually solved as a classification problem, we'll discuss the two metrics that would apply here: .

**Precision & Recall:** The accuracy of most classification algorithms are measured by their precision and recall. If you've been doing a lot of reading about machine learning, you may have come across these terms already. You need to know what they are, because they're what will ultimately get your prototype approved. Worry not, they're really simple.

Precision is simply the true positives that the algorithm was able to get on a particular test dataset. In other words, this percentage of correct answers. So, let's assume that it made 1000 predictions, and 850 of them were correct, precision would be 85%. The higher the number of right answers, the lower the number of false positives .

On the other hand, recall measures the ability of your model to correctly predict the true positives. Therefore, whereas precision takes into account the

ability to get all predictions correct, recall only measures how many true positives were accurately guessed. So, let's assume that there's a model is required to predict whether or not a person is likely to purchase a particular item. If, in a test dataset of 1000, it said that only 300 would purchase the product and, but 75 of the 300 that it guessed were false positives, the recall would be 75% (300-75*100/300). Put differently, the higher your recall rate, the fewer the number of false negatives.

In all classification models, you will need to make a tradeoff between precision and recall. If your model relies on precision, it will have more false negatives (or lower recall). On the other hand, if your model relies on recall, it will have more false positives (or higher recall) as it would be better to label something as a positive than incorrectly classify it as a negative. This is, of course, a business decision. In the case of fraud detection, we would usually want a higher recall, as it's absolutely vital that we flag off all suspicious transactions.

| n=165 | **Predicted: NO** | **Predicted: YES** | |
|---|---|---|---|
| **Actual: NO** | TN = 50 | FP = 10 | 60 |
| **Actual: YES** | FN = 5 | TP = 100 | 105 |
| | 55 | 110 | |

*Confusion matrix: Typically, in a supervised learning model, the error rates are visualised in a confusion matrix, (or an error matrix).*

**Imbalanced training data:** When calculating precision and recall, though, it's important to ensure that the data is properly balanced. Let's say you've fed

your model a thousands of instances of fraud, but they're of the same type; however, there may be 100s of other types for which you have only a few data points. What happens then? When you calculate overall precision/recall, the numbers may look good. They will, however, mask all those areas that your model is not able to spot due to a lack of data. As the model we're building is extremely important to the bank and its customers, it would be necessary to ensure that the model has been fed enough data points on all types.

Your model can move to the next (and final) stage when you're convinced that it is ready for the real world. Of course, there's still lots to figure out. But typically, the productization step is when the model is in beta.

## 6. Moving it to production

For the first time, your product will be away from the watch of the data science team. You will want to scale the product in a staggered manner, first testing it with a small sample of people, before opening it up to larger volumes. You can decide the scale that is required. You may decide to open a section of the model to all users or the entire product to only a few users. Depending on what you choose to do, there will be other considerations:

**Scaling your model:** Once you send your model out into the real world -- and particularly if you expose it to a large dataset in real-time -- it's likely that you will need another training and testing phase. Models don't usually cope well with an entirely new dataset. At this stage, the data engineering team would also need to architect the model in such a manner that it holds structurally when presented with vast amounts of data (which it must process and also memorise for future use).

**Increasing coverage of data:** Your model has been trained and tested on a limited dataset. Even if you choose to enter into beta with only a small section of users, the data that the model would be exposed to would be much larger. For this to even be possible, you will need to refine the data gathering process to the point that it's all automated. If, for example, you're scraping some government data or looking up some information on a website, all of this needs to happen in real-time as a transaction is taking place.

While you may have done all your data gathering in an ad-hoc manner earlier, it would all need to be seamless if this has to be rolled out to the real world. Moreover, you would also need to decide which data needs to be stored and what can be discarded. This is because your system will be bombarded with growing quantities of data as it covers more and more people. You will need to take a call on whether the data should be stored or discarded.

**Adjust for outliers:** You will also need to keep track of certain outliers for which the model doesn't work very well. For example, people who work the night shift and shop heavily online at 3am or pursers who fly across the globe but use their local card. You will need to find a solution because it would be disastrous if you had all their transactions cancelled. You may need to put in place a temporary solution (have the call centre be alerted so that they can speak to the customer) before you figure out a way for your model to understand these cases.

The building of a model requires numerous iterations. There is no end to adjusting for outliers or automating data collection or adding useful data points. You will simply have to refine it as per the requirements of the organisation. But if you've done all that we've discussed above, you've gotten off to a very good start.

Do note, of course, that what we've discussed is a very rough sketch of how it actually works. In reality, and even this will vary from one project to the next, you may spend a lot of time simply at the data gathering and cleaning stage or keep looping from prototyping to ideation because of unsatisfactory results. There will be a lot of rethinking and a lot of rework. But it's a fun, iterative process that is crucial to the organisation. Welcome to the world of data science!