Dealing with the Lack of Data in Machine Learning

Alexandre Gonfalonieri Follow

May 17, 2019 · 10 min read ★

In many projects I carried out, companies, despite having fantastic AI business ideas, display a tendency to slowly become frustrated when they realize that they do not have enough data… However, solutions do exist! **The purpose of this article is to briefly**

introduce you to some of them (the ones that are proven effective in my practice) rather than to list all existing solutions.

The problem of data scarcity is very important since data are at the core of any AI project. The size of a dataset is often responsible for poor performances in ML projects.

Most of the time, data related issues are the main reason why great AI projects cannot be accomplished. In some projects, you come to the conclusion that there is no relevant data or the collection process is too difficult and time-consuming.

Supervised machine learning models are being successfully used to respond to a whole range of business challenges. However, these models are data-hungry and their performance relies heavily on the size of training data available. In many cases, it is difficult to create training datasets that are large enough.

Another issue I could mention is that project analysts tend to underestimate the amount of data necessary to handle common business problems. I remember myself struggling to collect big training datasets. It is even more complicated to gather data when working for a large company…

How much data do I need?

Well, you need roughly 10 times as many examples as there are degrees of freedom in your model. The more complex the model, the more you are prone to overfitting, but that can be avoided by validation. **However, much fewer data can be used based on the use case.**

> ***Overfitting:*** *refers to a model that models the training data too well. It happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data.*

It is also worth discussing the issue of handling the missing values. Especially, if the number of missing values in your data is big enough (above 5%).

Once again, dealing with missing values will depend on certain 'success' criteria. Moreover, these criteria vary for different datasets and even for different applications such as recognition, segmentation, prediction, classification, etc. (given the same dataset) even for different applications (recognition, segmentation, prediction, classification).

> It is important to understand that there is no perfect way to deal with missing data.

Different solutions exist but it depends on the kind of problem — Time-series Analysis, ML, Regression, etc.

When it comes to predictive techniques, they shall be used only when missing values are not observed completely at random and the variables were chosen to impute such missing values have some relationship with it, else it could yield imprecise estimates.

In general, different machine learning algorithms can be used to determine the missing values. This works by turning missing features to labels themselves and now using columns without missing values to predict columns with missing values.

Based on my experience, you will be confronted with a lack of data or missing data at some point if you decide to build an AI-powered solution, **but fortunately, there are ways to turn that minus into a plus.**

Lack of data?

As noted above, it is impossible to precisely estimate the minimum amount of data required for an AI project. Obviously, the very nature of your project will influence significantly the amount of data you will need. For example, texts, images, and videos usually require more data. **However, many other factors should be considered in order to make an accurate estimate.**

- **Number of categories to be predicted**
  What is the expected output of your model? Basically, the fewest number or categories the better.

- **Model Performance**
  If you plan on getting a product in production, you need more. **A small dataset**

> **might be good enough for a proof of concept but in production, you'll need way more data.**

In general, small datasets require models that have low complexity (or high bias) to avoid overfitting the model to the data.

Non-Technical Solutions

Before exploring technical solutions, let's analyze what we can do to enhance your dataset. It might sound obvious but before getting started with AI, please try to obtain as much data as possible by developing your external and internal tools with data collection in mind. If you know the tasks that a machine learning algorithm is expected to perform, you can create a data-gathering mechanism in advance.

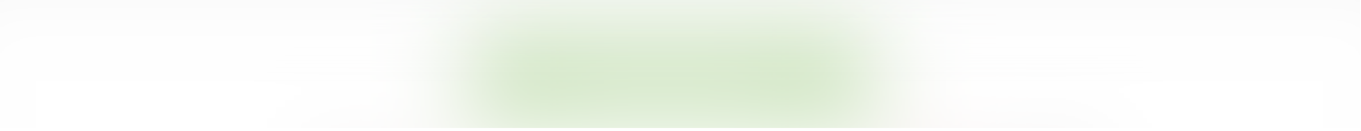
> Try to establish a real data culture within your organization.

To initiate ML execution, you could rely on open source data. There are a lot of data available for ML and some companies are ready to give it away.

If you need external data for your project, it can be beneficial to form partnerships with other organizations in order to get relevant data. Forming partnerships will cost obviously cost you some time, but the proprietary data gained will build a natural barrier to any rivals.

Build a useful application, give it away, use the data

Another approach that I used in my previous project was to give away access to a cloud application to customers. The data that makes it into the app can be used to build machine learning models. My previous client built an application for hospitals and made it free. We gathered a lot of data thanks to it and managed to create a unique dataset for our ML solution. It really helps to tell customers or investor that you have built your own and unique dataset.



Small datasets

Based on my experience, some common approaches that can help with building predictive models from small data sets are:

In general, the simpler the machine learning algorithm the better it will learn from small data sets. From an ML perspective, **small** data requires models that have low complexity (or high bias) to avoid overfitting the model to the data. I noticed that the Naive Bayes algorithm is among the simplest classifiers and as a result learns remarkably well from relatively small data sets.

> ***Naive Bayes methods:*** *set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable.*

You can also rely on other linear models and decision trees. Indeed, they can also perform relatively well on small data sets. Basically, simple models are able to learn from small data sets better than more complicated models (neural networks) since they are essentially trying to learn less.

For very **small datasets**, Bayesian methods are generally the best in class, although the results can be sensitive **to your choice of prior.** I think that the naive Bayes classifier and ridge regression are the best predictive models.

When it comes to **small** datasets, you need models that have few parameters (low complexity) and/or a strong prior. You can also interpret the "prior" as an assumption you can make on how the data behaves.

Many other solutions do exist depending on the exact nature of your business issues and the size of your dataset.

Transfer learning

> ***Definition:*** *a framework that leverages existing relevant data or models while building a machine learning model.*

Transfer learning uses knowledge from a learned task to improve the performance on a related task, typically reducing the amount of required training data.

Transfer learning techniques are useful because they allow models to make predictions for a new domain or task (known as the target domain) using knowledge learned from another dataset or existing machine learning models (the source domain).

Transfer learning techniques should be considered when you do not have enough target training data, and the source and target domains have some similarities but are not identical.

Naively aggregating models or different datasets would not always work! If the existing datasets are very different from the target data then the new learner can be negatively impacted by existing data or models.

Transfer learning works well when you have other datasets you can use to infer knowledge, but what happens when you have no data at all? This is where data generation can play a role. It is used when no data is available, or when you need to create more data than you could amass even through aggregation.

In this case, the small amount of data that does exist is modified to create variations on that data to train the model. For example, many images of a car can be generated by cropping, cropping, downsizing, one single image of a car.

Unfortunately, the lack of quality labeled data is also one of the largest challenges facing data science teams, but by using techniques such as transfer learning and data generation it is possible to overcome data scarcity.

Another common application of transfer learning is to train models on cross-customer datasets to overcome the cold-start problem. I noticed that SaaS companies often have to deal with when onboarding new customers to their ML products. Indeed, until the new customer has collected enough data to achieve good model performance (which could take several months) it's hard to provide value

Data Augmentation

Data augmentation means increasing the number of data points. In my latest project, we used data augmentation techniques to increase the number of images in our dataset. In terms of traditional row/column format data, it means increasing the number of rows or objects.

We had no choice but to rely on data augmentation for two reasons: Time and Accuracy. Every data collection process is associated with a cost. This cost can be in terms of dollars, human effort, computational resources and of course time consumed in the process.

As a consequence, we had to augment existing data to increase the data size that we feed to our ML classifiers and to compensate for the cost involved in further data collection.

> There are many ways to augment data.

In our case, you can rotate the original image, change lighting conditions, crop it differently, so for one image you can generate different sub-samples. **This way you can reduce overfitting your classifier.**

However, if you are generating artificial data using over-sampling methods such as SMOTE, then there is a fair chance you may introduce over-fitting.

> ***Over-fitting:*** *An overfitted model is a model with a trend line that reflects the errors in the data that it is trained with, instead of accurately predicting unseen data.*

This is something you must take into consideration when developing your AI solution.

Synthetic Data

Synthetic data means fake data that contains the same schema and statistical properties as its "real" counterpart. Basically, it looks so real that it's nearly impossible to tell that it's not.

So what's the point of synthetic data, and why does it matter if we already have access to the real thing?

I have seen synthetic data applied especially when we were dealing with private data (banking, healthcare, etc.), this makes the use of synthetic data a more secure approach to development in certain instances.

Synthetic data is used mostly when there is not enough real data or there is not enough real data for specific patterns you know about. Usage mostly the same for training and testing datasets.

Synthetic Minority Over-sampling Technique (SMOTE) and Modified- SMOTE are two such techniques which generate synthetic data. Simply put, SMOTE takes the minority class data points and creates new data points which lie between any two nearest data points joined by a straight line.

The algorithm calculates the distance between two data points in the feature space, multiplies the distance by a random number between 0 and 1 and places the new data point at this new distance from one of the data points used for distance calculation.

In order to generate synthetic data, you have to use a Training Set to define a model, which would require validation, and then by changing the parameters of interest, you can generate synthetic data, through simulation. The domain/data type is significant since it affects the complexity of the entire process.

In my opinion, asking yourself if you have enough data will reveal inconsistencies that you have probably never spotted before. It will help to highlight issues in your business processes that you thought were perfect and make you understand why it is the key to creating a successful data strategy within your organization.

Machine Learning
Artificial Intelligence
Data Science
Business
Technology