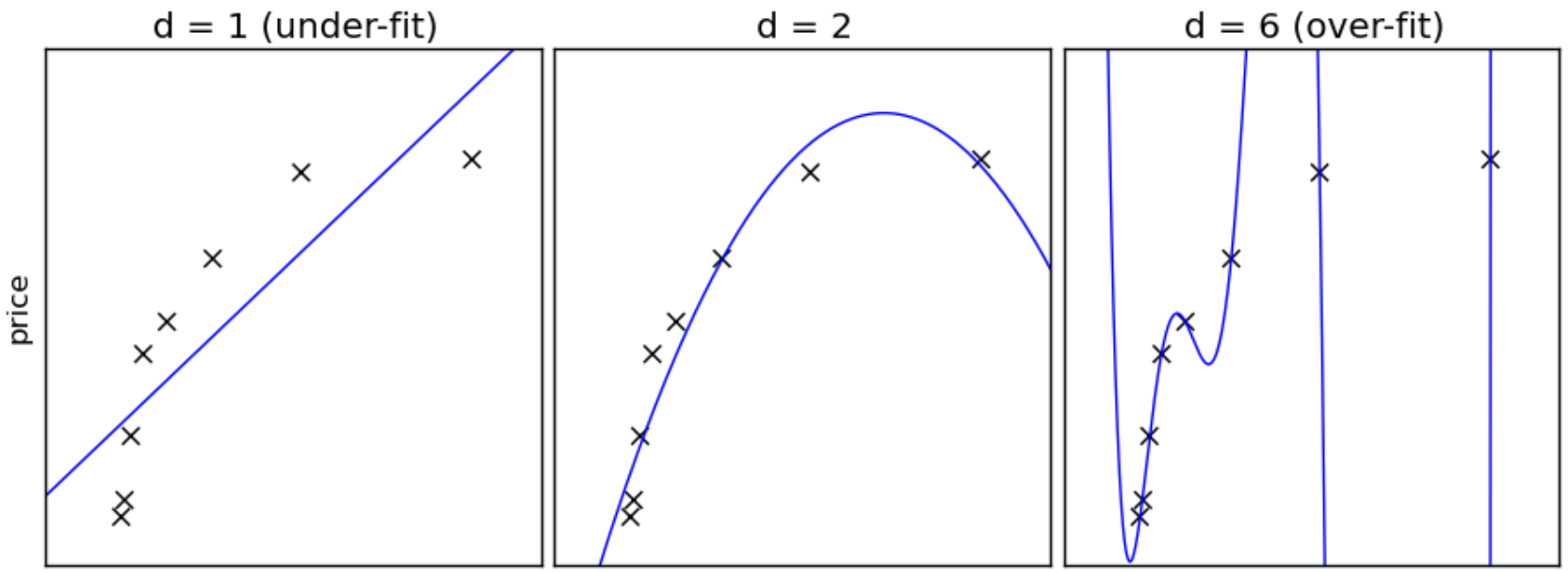
New to Machine Learning? Avoid these three mistakes

Common pitfalls when learning from data

Nov 8, 2013

@nomadic_mind. Sometimes the difference between success and failure is the same as between = and ==. Living is in the details.



Machine learning (ML) is one of the hottest fields in data science. As soon as ML entered the mainstream through Amazon, Netflix, and Facebook people have been giddy about what they can learn from their data. However, modern machine learning (*i.e.* not the theoretical statistical learning that emerged in the 70s) is very much an evolving field and despite its many successes we are still learning what exactly can ML do for data practitioners. I gave a talk on this topic earlier this fall at Northwestern University and I wanted to share these cautionary tales with a wider audience.

Machine learning is a field of computer science where algorithms improve their performance at a certain task as more data are observed.To do so, algorithms select a hypothesis that best explains the data at hand with the hope that the hypothesis would *generalize* to future (unseen) data. Take

the left panel in the figure in the header, the crosses denote the observed data projected in a two-dimensional space—in this case house prices and their corresponding size in square meters. The blue line is the algorithm's best hypothesis to explain the observed data. It states "there is a linear relationship between the price and size of a house. As the house's size increases, so does its price in linear increments." Now using this hypothesis, I can predict the price of an unseen datapoint based on its size. As the dimensions of the data increase, the hypotheses that explain the data become more complex.However, given that we are using a finite sample of observations to learn our hypothesis, finding an adequate hypothesis that generalizes to unseen data is nontrivial. There are three major pitfalls one can fall into that will prevent you from having a generalizable model and hence the conclusions of your hypothesis will be in doubt.

Occam's Razor

Occam's razor is a principle attributed to William of Occam a 14th century philosopher. Occam's razor advocates for choosing the simplest hypothesis that explains your data, yet no simpler. While this notion is simple and elegant, it is often misunderstood to mean that we must select the simplest hypothesis possible regardless of performance.

> *The simplest hypothesis that fits the data is also the most plausible*

In their 2008 paper in *Nature*, Johan Nyberg and colleagues used a 4-level artificial neural network to predict seasonal hurricane counts using two or three environmental variables. The authors reported stellar accuracy in predicting seasonal North Atlantic hurricane counts, however their model violates Occam's razor and most certainly doesn't generalize to unseen data. The razor was violated when the hypothesis or model selected to describe the relationship between environmental data and seasonal hurricane counts was generated using a four-layer neural network. A four-

layer neural network can model virtually any function no matter how complex and could fit a small dataset very well but fail to generalize to unseen data. The rightmost panel in the top figure shows such incident. The hypothesis selected by the algorithm (the blue curve) to explain the data is so complex that it fits through every single data point. That is: for any given house size in the training data, I can give you with pinpoint accuracy the price it would sell for. It doesn't take much to observe that even a human couldn't be that accurate. We could give you a very close estimate of the price, but to predict the selling price of a house, within a few dollars , every single time is impossible.

> *Think of overfitting as memorizing as opposed to learning.*

The pitfall of selecting too complex a hypothesis is known as *overfitting*. Think of overfitting as memorizing as opposed to learning. If you are a child and you are memorizing how to add numbers you may memorize the sums of any pair of integers between 0 and 10. However, when asked to calculate 11 + 12 you will be unable to because you have never seen 11 or 12, and therefore couldn't memorize their sum. That's what happens to an overfitted model, it gets too lazy to learn the general principle that explains the data and instead memorizes the data.

Data leakage

Data leakage occurs when the data you are using to learn a hypothesis happens to have the information you are trying to predict. The most basic form of data leakage would be to use the same data that we want to predict as input to our model (*e.g.* use the price of a house to predict the price of the same house). However, most often data leakage occurs subtly and inadvertently. For example, one may wish to learn for anomalies as opposed to raw data, that is a deviations from a long-term mean. However, many fail to remove the test data *before* computing the anomalies and hence the anomalies carry some information about the data you want to

predict since they influenced the mean and standard deviation before being removed.

The are several ways to avoid data leakage as outlined by Claudia Perlich in her great paper on the subject. However, there is no silver bullet — sometimes you may inherit a corrupt dataset without even realizing it. One way to spot data leakage is if you are doing very poorly on unseen independent data. For example, say you got a dataset from someone that spanned 2000-2010, but you started collecting you own data from 2011 onward. If your model's performance is poor on the newly collected data it may be a sign of data leakage. You must resist the urge to retrain the model with both the potentially corrupt and new data. Instated, either try to identify the causes of poor performance on the new data or, better yet, independently reconstruct the entire dataset. As a rule of thumb, your best defense is to always be mindful of the possibility of data leakage in any dataset.

> *For every Apple that became a success there were 1000 other startups that died trying.*

Sampling Bias

Sampling bias is the case when you shortchange your model by training it on a biased or non-random dataset, which results in a poorly generalizable hypothesis. In the case of housing prices, sampling bias occurs if, for some reason, all the house prices/sizes you collected were of huge mansions. However, when it was time to test your model and the first price you needed to predict was that of a 2-bedroom apartment you couldn't predict it. Sampling bias happens very frequently mainly because, as humans, we are notorious for being biased (nonrandom) samplers. One of the most common examples of this bias happens in startups and investing. If you attend any business school course, they will use all these "case studies" of how to build a successful company. Such case studies actually depict the anomalies and not the norm as most companies fail — For every Apple that

became a success there were 1000 other startups that died trying. So to build an automated data-driven investment strategy you would need samples from both successful and unsuccessful companies.

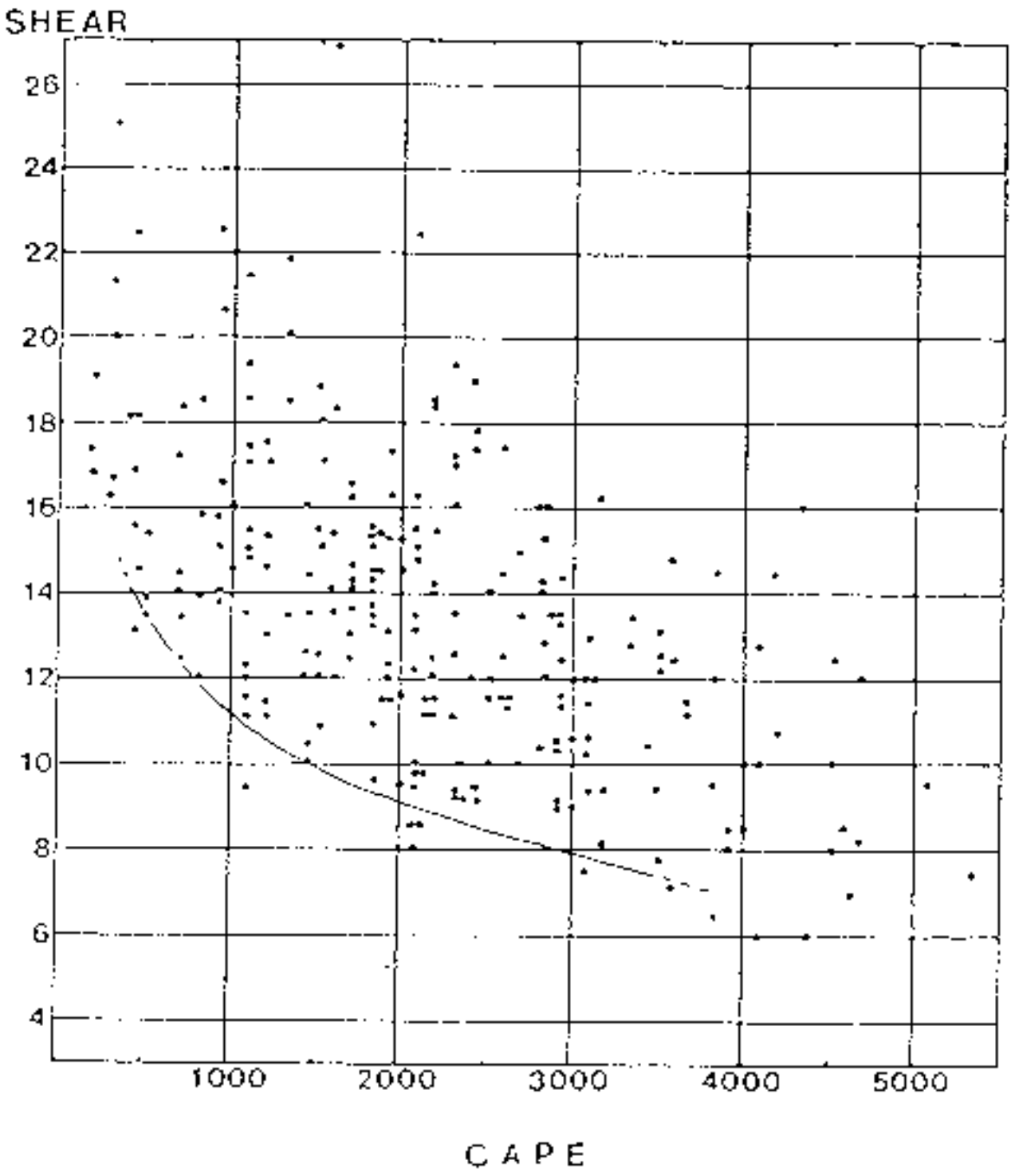


Figure 13. Scatter diagram showing the relationship between convectively available potential energy (CAPE) and 0-2 km AGL wind shear (X $10^{-3}s^{-1}$) for 242 tornado cases (Johns, et al., 1990).

This plot tries to infer the environmental conditions that lead to tornado genesis. This is an example of sampling bias, because the authors only recorded the environmental conditions (SHEAR and CAPE) for tornado occurrences but not for instances when a tornado did not originate. In order to build a generalizable model on must include samples from all populations within the data (i.e. positive and negative events)

The figure above (Figure 13) is a concrete example of sampling bias. Say you want to predict whether a tornado is going to originate at certain location based on two environmental conditions: wind shear and convective available potential energy (CAPE). We don't have to worry about what these variables actually mean, but Figure 13 shows the wind shear and CAPE associated with 242 tornado cases. We can fit a model to these data but it will certainly not generalize because we failed to include shear and CAPE values when tornados *did not occur*. In order for our model to separate between positive (tornados) and negative (no tornados) events we must train it using both populations.

There you have it. Being mindful of these limitations does not guarantee that your ML algorithm will solve all your problems, but it certainly reduces the risk of being disappointed when your model doesn't generalize to unseen data. Now go on young Jedi: train your model, you must!

> ***If you found value in this article****, I would be grateful if you hit the recommend or Tweet button below.*

Write a response…

Ji Hyeon Hyeong