Data Science essentials: Why train-validation-test data?

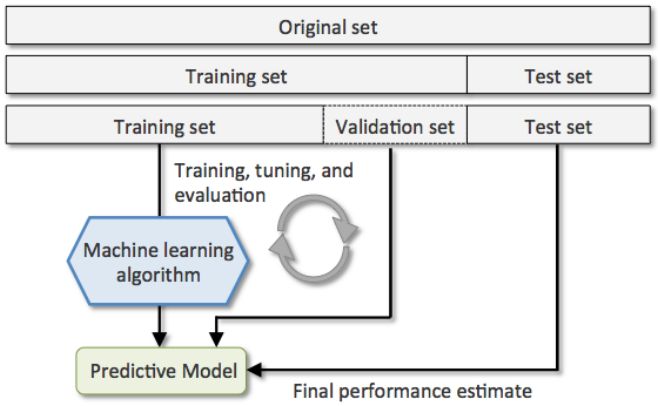
Sagar Patel Follow
Sep 24, 2018 · 3 min read

Ever wondered why we split the data into train-validation-test?

Here is the table that sums it all

	Purpose	Yield	Used for Model training	Used for Parameter tuning
Train Data	To learn patterns from the data.	A model that makes near-expected predictions	Yes	Yes
Validation Data	To understand model behaviour and generalizibility on unseen data.	Insights on how to tune your model.	No	Yes
Test Data	To understand how the model would perform in real world scenario.	A completely unbiased estimate of model performance.	No	No

This is how the architecture looks like…



Important points to note…

1. **Validation data is a part of iterative loop.** We frequently take insights from validation errors to tune our models. So we are **implicitly leaking information from our validation data to our model.**

2. **Advanced validation methods have obscured the importance of single split validation data.** K-fold cross-validation is quite robust and probably the current industry standard for model performance validation and parameter tuning. **So if you are using cross-validation techniques in your analysis, you may ignore the validation data split.**

3. The primary objective of test data is to give an unbiased estimate of model accuracy. It should be used at the very end and only for a couple of times. **If you tune your model after looking at the test accuracies, you are technically leaking information and hence cheating.**

4. For the very same reason as above *(leakage of information)*, in spite of the programming convenience we should not combine train-validation-test dataset to make common preprocessing flow. Some might argue that according to the base hypothesis train-validation-test data come from the same population distribution and hence there should be no harm in combining them for a common preprocessing flow. This is true in idealistic scenarios, but real life if far from it as you never know when your real-time production system start getting evolving data (*whose distribution is slightly different from the training data)*. **As a good Data Scientist you should strive to make a model flow that is generalizable and performs well *(without any additional changes)*irrespective of the uncertainties in future data.**

5. We are suppose to develop 2 separate preprocessing pipelines. (A) for training data and (B) for validation and test data. However it should be noted that these pipelines aren't completely independent. **You learn the transformation features *(mean/range/standard-deviation)* from training data and use it to transform your validation and test data.**

6. And finally the most important thing *(at the cost of sounding silly)* **never make the mistake of training your model on validation or test data.** When I code for my projects, I make sure my validation and test data frames stay away from model.fit() function. >_<

As a consultant Data Scientist and an academic trainer I've heard numerous stories where the lack of understanding on the above points have turned into project disasters. Hope this article helps you avoid those pitfalls. If you have anything to add or correct feel free to comment.