Beyond Bayesians and Frequentists

Jacob Steinhardt

October 31, 2012

If you are a newly initiated student into the field of machine learning, it won't be long before you start hearing the words "Bayesian" and "frequentist" thrown around. Many people around you probably have strong opinions on which is the "right" way to do statistics, and within a year you've probably developed your own strong opinions (which are suspiciously similar to those of the people around you, despite there being a much greater variance of opinion between different labs). In fact, now that the year is 2012 the majority of new graduate students are being raised as Bayesians (at least in the U.S.) with frequentists thought of as stodgy emeritus professors stuck in their ways.

If you are like me, the preceding set of facts will make you very uneasy. They will make you uneasy because simple pattern-matching – the strength of people's opinions, the reliability with which these opinions split along age boundaries and lab boundaries, and the ridicule that each side levels at the other camp makes the "Bayesians vs. frequentists" debate look far more like politics than like scholarly discourse. Of course, that alone does not necessarily prove anything; these disconcerting similarities could just be coincidences that I happened to cherry-pick.

My next point, then, is that we are right to be uneasy, because such debate makes us less likely to evaluate the strengths and weaknesses of both approaches in good faith. This essay is a push against that — I summarize the justifications for Bayesian methods and where they fall short, show how frequentist approaches can fill in some of their shortcomings, and then present my personal (though probably woefully under-informed) guidelines for choosing which type of approach to use.

Before doing any of this, though, a bit of background is in order...

1 Background on Bayesians and Frequentists

1.1 Three Levels of Argument

As Andrew Critch [6] insightfully points out, the Bayesians vs. frequentists debate is really three debates at once, centering around one or more of the following arguments:

1. Whether to interpret subjective beliefs as probabilities

2. Whether to interpret probabilities as subjective beliefs (as opposed to asymptotic frequencies)

3. Whether a Bayesian or frequentist algorithm is better suited to solving a particular problem.

Given my own research interests, I will add a fourth argument:

4. Whether Bayesian or frequentist techniques are better suited to engineering an artificial intelligence.

Andrew Gelman [9] has his own well-written essay on the subject, where he expands on these distinctions and presents his own more nuanced view.

Why are these arguments so commonly conflated? I'm not entirely sure; I would guess it is for historical reasons but I have so far been unable to find said historical reasons. Whatever the reasons, what this boils down to in the present day is that people often form opinions on 1. and 2., which then influence their answers to 3. and 4. This is *not good*, since 1. and 2. are philosophical in nature and difficult to resolve correctly, whereas 3. and 4. are often much easier to resolve and extremely important to resolve correctly in practice. Let me re-iterate: *the Bayes vs. frequentist discussion should center on the practical employment of the two methods, or, if epistemology must be discussed, it should be clearly separated from the day-to-day practical decisions.* Aside from the difficulties with correctly deciding epistemology, the relationship between generic epistemology and specific practices in cutting-edge statistical research is only via a long causal chain, and it should be completely unsurprising if Bayesian epistemology leads to the employment of frequentist tools or vice versa.

For this reason and for reasons of space, I will spend the remainder of the essay focusing on *statistical algorithms* rather than on *interpretations of probability*. For those who really want to discuss interpretations of probability, I will address that in a later essay.

1.2 Recap of Bayesian Decision Theory

(What follows will be review for many.) In Bayesian decision theory, we assume that there is some underlying world state θ and a *likelihood function* $p(X_1, \dots, Xn \mid \theta)$ over possible observations[1]. We also have a space A of possible actions and a utility function $U(\theta; a)$ that gives the utility of performing action a if the underlying world state is θ. We can incorporate notions like planning and value of information by defining $U(\theta; a)$ recursively in terms of an identical agent to ourselves who has seen one additional observation (or, if we are planning against an adversary, in terms of the adversary). For a more detailed overview of this material, see the tutorial by North [11].

What distinguishes the Bayesian approach in particular is one additional assumption, a *prior distribution* $p(\theta)$ over possible world states. To make a decision with respect to a given prior, we compute the posterior distribution $p_{\mathrm{posterior}}(\theta \mid X_1, \dots, X_n)$ using Bayes' theorem, then take the action a that maximizes $\mathbb{E}_{p_{\mathrm{posterior}}}[U(\theta; a)]$.

In practice, $p_{\mathrm{posterior}}$ can be quite difficult to compute, and so we often attempt to approximate it. Such attempts are known as *approximate inference algorithms*.

1.3 Steel-manning Frequentists

There are many different ideas that fall under the broad umbrella of frequentist techniques. While it would be impossible to adequately summarize all of them even if I attempted to, there are three in particular that I would like to describe, and which I will call *frequentist decision theory*, *frequentist guarantees*, and *frequentist analysis tools*.

Frequentist decision theory has a very similar setup to Bayesian decision theory, with a few key differences. These are discussed in detail and contrasted with Bayesian decision theory in [10], although we summarize the differences here. There is still a likelihood function $p(X_1, \dots, X_n | \theta)$ and a utility function $U(\theta; a)$. However, we do not assume the existence of a prior on θ, and instead choose the decision rule $a(X_1, \dots, X_n)$ that maximizes

[1] A *likelihood function* is just a conditional probability distribution where the parameter conditioned on can vary.

$$\min_{\theta} \mathbb{E}[U(a(X_1, \dots, X_n); \theta) \mid \theta]. \tag{1}$$

In other words, we ask for a worst case guarantee rather than an average case guarantee[2]. Note that this is more complicated than in the Bayesian case, since the value of (1) depends on the joint behavior of $a(X_1, \dots, X_n)$, whereas with Bayes we can optimize $a(X_1, \dots, X_n)$ for each set of observations separately.

As a result of this more complex optimization problem, it is often not actually possible to maximize (1), so many frequentist techniques instead develop tools to lower-bound (1) for a given decision procedure, and then try to construct a decision procedure that is reasonably close to the optimum. Support vector machines [2], which try to pick separating hyperplanes that minimize generalization error, are one example of this where the algorithm is explicitly trying to maximize worst-case utility. Another example of a frequentist decision procedure is L1-regularized least squares for sparse recovery [3], where the procedure itself does not look like it is explicitly maximizing any utility function, but a separate analysis shows that it is close to the optimal procedure anyways.

The second sort of frequentist approach to statistics is what I call a *frequentist guarantee*. A frequentist guarantee on an algorithm is a guarantee that, with high probability with respect to how the data was generated, the output of the algorithm will satisfy a given property. The most familiar example of this is any algorithm that generates a frequentist confidence interval: to generate a 95% frequentist confidence interval for a parameter θ is to run an algorithm that outputs an interval, such that with probability at least 95% θ lies within the interval. An important fact about most such algorithms is that the size of the interval only grows logarithmically with the amount of confidence we require, so getting a 99.9999% confidence interval is only slightly harder than getting a 95% confidence interval (and we should probably be asking for the former whenever possible).

If we use such algorithms to test hypotheses or to test discrete properties of θ, then we can obtain algorithms that take in probabilistically generated data and produce an output that with high probability depends *only on how the data was generated*, not on the specific random samples that were given. For instance, we can create an algorithm that takes in samples from two distributions, and is guaranteed to output 1 whenever they are the same, 0 whenever they differ by at least ϵ in total variational distance, and could have arbitrary output if they are different but the total variational distance is less than ϵ. This is an amazing property — it takes in random input and produces an essentially deterministic answer.

Finally, a third type of frequentist approach seeks to construct *analysis tools* for understanding the behavior of random variables. Metric entropy, the Chernoff and Azuma-Hoeffding bounds [12], and Doob's optional stopping theorem are representative examples of this sort of approach. Arguably, everyone with the time to spare should master these techniques, since being able to analyze random variables is important no matter what approach to statistics you take. Indeed, frequentist analysis tools have no conflict at all with Bayesian methods — they simply provide techniques for understanding the behavior of the Bayesian model.

[2] As an example of how these would differ, imagine a scenario where we have no data to observe, an unknown $\theta \in \{1, \dots, N\}$, and we choose an action $a \in \{0, \dots, N\}$. Furthermore, $U(0; \theta) = 0$ for all θ, $U(a; \theta) = -1$ if $a = \theta$, and $U(a; \theta) = 1$ if $a \neq 0$ and $a \neq \theta$. Then a frequentist will always choose $a = 0$ because any other action gets -1 utility in the worst case; a Bayesian, on the other hand, will happily choose any non-zero value of a since such an action gains $\frac{N-2}{N}$ utility in expectation. (I am purposely ignoring more complex ideas like mixed strategies for the purpose of illustration.)

2 Bayes vs. Other Methods

2.1 Justification for Bayes

We presented Bayesian decision theory above, but are there any reasons why we should actually use it? One commonly-given reason is that Bayesian statistics is merely the application of Bayes' Theorem, which, being a theorem, describes the only correct way to update beliefs in response to new evidence; anything else can only be justified to the extent that it provides a good approximation to Bayesian updating. This may be true, but Bayes' Theorem only applies if we already have a prior, and if we accept probability as the correct framework for expressing uncertain beliefs. We might want to avoid one or both of these assumptions. Bayes' theorem also doesn't explain why we care about expected utility as opposed to some other statistic of the distribution over utilities (although note that frequentist decision theory also tries to maximize expected utility).

One compelling answer to this is **Cox's Theorem**, which shows that any agent must implicitly be using a probability model to make decisions, or else they can be *dutch-booked* — meaning there is a series of bets that they would be willing to make that causes them to lose money with certainty. Another answer is the **complete class theorem**, which shows that any non-Bayesian decision procedure is *strictly dominated* by a Bayesian decision procedure — meaning that the Bayesian procedure performs at least as well as the non-Bayesian procedure in all cases with certainty. In other words, if you are doing anything non-Bayesian, then either it is secretly a Bayesian procedure or there is another procedure that does strictly better than it. Finally, the **VNM Utility Theorem** states that any agent with consistent preferences over distributions of outcomes must be implicitly maximizing the expected value of some scalar-valued function, which we can then use as our choice of utility function U. These theorems, however, ignore the issue of computation — while the best decision procedure may be Bayesian, the best computationally-efficient decision procedure could easily be non-Bayesian.

Another justification for Bayes is that, in contrast to ad hoc frequentist techniques, it actually provides a general theory for constructing statistical algorithms, as well as for incorporating side information such as expert knowledge. Indeed, when trying to model complex and highly structured situations it is difficult to obtain any sort of frequentist guarantees (although analysis tools can still often be applied to gain intuition about parts of the model). A prior lets us write down the sorts of models that would allow us to capture structured situations (for instance, when trying to do language modeling or transfer learning). Non-Bayesian methods exist for these situations, but they are often ad hoc and in many cases ends up looking like an approximation to Bayes. One example of this is Kneser-Ney smoothing for n-gram models, an ad hoc algorithm that ended up being very similar to an approximate inference algorithm for the hierarchical Pitman-Yor process [16, 15, 18, 8]. This raises another important point *against* Bayes, which is that the proper Bayesian interpretation may be very mathematically complex. Pitman-Yor processes are on the cutting-edge of Bayesian nonparametric statistics, which is itself one of the more technical subfields of statistical machine learning, so it was probably much easier to come up with Kneser-Ney smoothing than to find the interpretation in terms of Pitman-Yor processes.

2.2 When the Justifications Fail

The first and most common objection to Bayes is that a Bayesian method is only as good as its prior. While for simple models the performance of Bayes is relatively independent of the prior, such models can only capture data where frequentist techniques would also perform very well. For more complex (especially nonparametric) Bayesian models, the performance can depend strongly on the prior, and designing good priors is still an open problem. As one example I point to my own research on hierarchical nonparametric models, where the most straightforward attempts to build a hierarchical model lead to severe pathologies [14].

Even if a Bayesian model does have a good prior, it may be computationally intractable to perform posterior inference. For instance, structure learning in Bayesian networks is NP-hard [4], as is topic inference in the popular latent Dirichlet allocation model (and this continues to hold even if we only want to perform approximate inference) [13]. Similar stories probably hold for other common models, although a theoretical survey has yet to be made; suffice to say that in practice approximate inference remains a difficult and unsolved problem, with many models not even considered because of the apparent hopelessness of performing inference in them.

Because frequentist methods often come with an analysis of the specific algorithm being employed, they can sometimes overcome these computational issues. One example of this mentioned already is L1 regularized least squares [3]. The problem setup is that we have a linear regression task $Ax = b + v$ where A and b are known, v is a noise vector, and x is believed to be sparse (typically x has many more rows than b, so without the sparsity assumption x would be underdetermined). Let us suppose that x has n rows and k non-zero rows — then the number of possible sparsity patterns is $\binom{n}{k}$ — large enough that a brute force consideration of all possible sparsity patterns is intractable. However, we can show that solving a certain semidefinite program[3] [17] will with high probability yield the appropriate sparsity pattern, after which recovering x reduces to a simple least squares problem.

Finally, Bayes has no good way of dealing with adversaries or with cases where the data was generated in a complicated way that could make it highly biased (for instance, as the output of an optimization procedure). A toy example of an adversary would be playing rock-paper-scissors — how should a Bayesian play such a game? The straightforward answer is to build up a model of the opponent based on their plays so far, and then to make the play that maximizes the expected score (probability of winning minus probability of losing). However, such a strategy fares poorly against any opponent with access to the model being used, as they can then just run the model themselves to predict the Bayesian's plays in advance, thereby winning every single time. In contrast, there is a frequentist strategy called the **multiplicative weights update method** that fairs well against an arbitrary opponent (even one with superior computational resources and access to our agent's source code). The multiplicative weights method does far more than winning at rock-paper-scissors — it is also a key component of the fastest algorithm for solving many important optimization problems (including the network flow algorithm), and it forms the theoretical basis for the widely used AdaBoost algorithm [1, 5, 7].

2.3 When To Use Each Method

The essential difference between Bayesian and frequentist decision theory is that Bayes makes the additional assumption of a prior over θ, and optimizes for average-case performance rather than worst-case performance. *It follows, then, that Bayes is the superior method whenever we can obtain a good prior and when good average-case performance is sufficient.* However, if we have no way of obtaining a good prior, or when we need guaranteed performance, frequentist methods are the way

[3] A *semidefinite program* is a certain type of optimization problem that can be solved efficiently.

to go. For instance, if we are trying to build a software package that should be widely deployable, we might want to use a frequentist method because users can be sure that the software will work as long as some number of easily-checkable assumptions are met.

A nice middle-ground between purely Bayesian and purely frequentist methods is to use a Bayesian model coupled with frequentist model-checking techniques; this gives us the freedom in modeling afforded by a prior but also gives us some degree of confidence that our model is correct. This approach is suggested by both Gelman [9] and Jordan [10].

3 Conclusion

When the assumptions of Bayes' Theorem hold, and when Bayesian updating can be performed computationally efficiently, then it is indeed tautological that Bayes is the optimal approach. Even when some of these assumptions fail, Bayes can still be a fruitful approach. However, by working under weaker (sometimes even adversarial) assumptions, frequentist approaches can perform well in very complicated domains even with fairly simple models; this is because, with fewer assumptions being made at the outset, less work has to be done to ensure that those assumptions are met.

From a research perspective, we should be far from satisfied with either approach — Bayesian methods make stronger assumptions than may be warranted, and frequentists methods provide little in the way of a coherent framework for constructing models, and ask for worst-case guarantees, which probably cannot be obtained in general. We should seek to develop a statistical modeling framework that, unlike Bayes, can deal with unknown priors, adversaries, and limited computational resources.

4 Acknowledgements

Thanks to Emma Pierson, Vladimir Slepnev, and Wei Dai for reading preliminary versions of this work and providing many helpful comments.

References

[1] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta algorithm and applications. *Working Paper*, 2005.

[2] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[3] Emmanuel J. Candès. Compressive sampling. In *Proceedings of the International Congress of Mathematicians*. European Mathematical Society, 2006.

[4] D.M. Chickering. Learning bayesian networks is np-complete. *LECTURE NOTES IN STATISTICS-NEW YORK-SPRINGER VERLAG-*, pages 121–130, 1996.

[5] Paul Christiano, Jonathan A. Kelner, Aleksander Madry, Daniel Spielman, and Shang-Hua Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *Proceedings of the 43rd ACM Symposium on Theory of Computing*, 2011.

[6] Andrew Critch. Frequentist vs. bayesian breakdown: Interpretation vs. inference. http://lesswrong.com/lw/7ck/frequentist_vs_bayesian_breakdown_interpretation/.

[7] Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, Sep. 1999.

[8] J. Gasthaus and Y. W Teh. Improvements to the sequence memoizer. In *Advances in Neural Information Processing Systems*, 2011.

[9] Andrew Gelman. Induction and deduction in bayesian data analysis. *RMM*, 2:67–78, 2011.

[10] Michael I. Jordan. Are you a bayesian or a frequentist? Machine Learning Summer School 2009 (video lecture at http://videolectures.net/mlss09uk_jordan_bfway/).

[11] D. Warner North. A tutorial introduction to decision theory. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(3):200–210, Sep. 1968.

[12] Igal Sason. On refined versions of the azuma-hoeffding inequality with applications in information theory. *CoRR*, abs/1111.1977, 2011.

[13] David Sontag and Daniel Roy. Complexity of inference in latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, 2011.

[14] Jacob Steinhardt and Zoubin Ghahramani. Pathological properties of deep bayesian hierarchies. In *NIPS Workshop on Bayesian Nonparametrics*, 2011. Extended Abstract.

[15] Y. W. Teh. A bayesian interpretation of interpolated kneser-ney. Technical Report TRA2/06, School of Computing, NUS, 2006.

[16] Y. W. Teh. A hierarchical bayesian language model based on pitman-yor processes. *Coling/ACL*, 2006.

[17] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, Mar. 1996.

[18] F. Wood, C. Archambeau, J. Gasthaus, L. James, and Y.W. Teh. A stochastic memoizer for sequence data. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1129–1136, 2009.