0.1 Introduction

The purpose of this note is to discuss the empirical measurement of difference between two random variables, as a tool for discussing hysteresis effects in the HPU outputs.

There are dozens (hundreds?) of quantities defined to measure the difference between the distributions of two random variables, many of which serve as metrics for spaces of distributions of the same dimension. We will keep our eye on two in particular.

The first, total variation (TV), offers the most straightforward measure and is easy to calculate. It also has an interpretation as the maximum bias introduced into a binary HPU computation "with binary loss of detail". The TV creates a link between classifier algorithms and random variable divergence. Based on this, any classifier is also a TV calculator.

The second, Jensen-Shannon divergence (JSD), will provide a natural information-theoretic interepretation in terms of the leakage of side-stream (non-input) information into HPU outputs.

0.2 HPUs

When we wish to speak of the computational properties of a person, we will refer to them as an HPU. In one central way, HPUs are similar to CPUs and GPUs: they perform computations. There are at lest three major differences between HPUs and their semiconductor counterparts: (a) HPUs are sentient entities, (b) HPUs have stochastic outputs, and (c) HPUs have ill-defined instruction sets.

The fact that HPUs are sentient entities doesn't have direct computational implications (unless it is seen as the cause of (b) and (c). However, it does mean that there is an important moral valence to HPU work. Some algorithms, when run on HPUs, might cause serious distress, which should be avoided.

This note will introduce tools for talking about (b), the stochasticity of HPUs, and we will take all the space below to expand on that. We will not touch further on (a) or (c).

Existing tools in probability information theory exist for discussing aspect (b). Here, we will advocate for using particular tools and definitions to characterize the stochastic nature of HPUs.

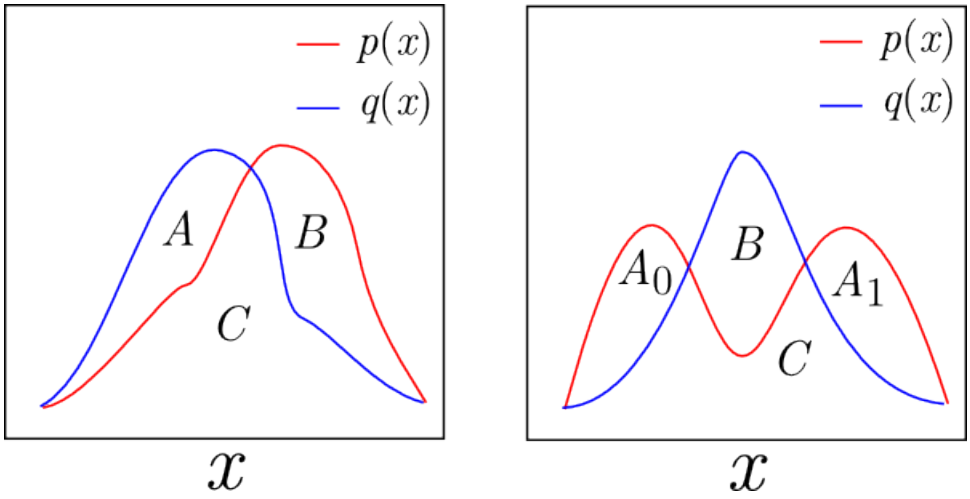
Imagine, for a moment, that the drive for better CPU or GPU performance took us into a situation where their behavior was stochastic. As miniaturization of transistors continued, at some point, quantum effects overwhelmed their predictable switching behavior. In that imaginary world, I would expect that there would be a stable probability distribution that would describe CPU behavior, and that it could be characterized, rather fully, experimentally.

HPUs, on the other hand, have a stochasticity that is in a certain sense "less random". They are not just stochastic – the probability distribution describing that stochastic behavior is affected by the state of the HPU, and can be influenced by priming. Think of this as a kind of "leakage of signal" from previous states or inputs into the current "frame".

For that reason, the majority of this note discusses how to compare two probability distributions, P and Q. We will make connections to HPU computation, but we will spend most of the time in the abstract. Keep in the back of your mind that P is the distribution over HPU outputs when HPUs were primed with one condition and Q is the same but when primed with another.

0.3 Total Variation

Consider the distributions shown in Fig. 0.3. In the first case, the area $A+B$ measures, in some sense the total difference between the distributions $p(x)$ and $q(x)$. A probability distribution must integrate, over



its support, to one, so $A + C = B + C \implies A = B$. For the second distribution, we get $A_1 + A_2 = B$. Thus, we only need to consider either the regions of the support where $p(x) > q(x)$, or those where $q(x) > p(x)$. This motivates the definition of total variation:

$$TV \equiv \sup_{S \subset X} \left| \sum_{x \in S} p(x) - q(x) \right| = \frac{1}{2} \sum_{x \in X} |p(x) - q(x)| \tag{1}$$

Naturally, Eq. 1 applies for discrete distributions, and should be replaced by an integral for continuous ones. For simplicity, we will always assume discrete distributions in this note.

0.4 Jensen-Shannon divergence (JSD)

The Jensen-Shannon divergence is a symmetrized version of the Kullback-Leibler divergence (D). For random variables P and Q, JSD is defined as:

$$JSD(P||Q) = \frac{1}{2} D(P||M) + \frac{1}{2} D(Q||M) \tag{2}$$

where M is a random variable from the mixed distribution $m(x) = \frac{1}{2}p(x) + \frac{1}{2}q(x)$, and where D stands for the Kullback-Leibler divergence, which, for discrete supports of P and Q, is given by:

$$D(P||M) = \sum_{x \in X} p(x) \lg \left(\frac{p(x)}{m(x)} \right) \tag{3}$$

JSD will have a special interpretation in the discussion of HPU outputs, but we will postpone comments about the meaning of this divergence measure until we have introduced the third divergence measure and discussed their mathematical relationships.

0.5 Algorithmic Distinguishability (θ)

The last measure of divergence is defined in terms of an an algorithm $\mathcal{D}$ which we will call a *distinguisher*. Generally one should think of a distinguisher simply as a classifier in the usual machine learning sense. I haven't found this definition elsewhere, so we'll build the this from scratch. First, let's formalize what we mean by a distinguisher:

Definition Let P and Q be random variables, and let the union of their supports be $\mathcal{X}$. Any algorithm $\mathcal{D}$ which accepts $x \in_{\mathrm{R}} \mathcal{X}$ as input and yields $z \in 0, 1$ as output is called a *distinguisher for P and Q*. The output, z, is called a *guess*.

We can think of z as the algorithm's attempt to guess whether the x it was shown was a sample from P or Q. Note that it isn't important whether a given algorithm is *good* at distinguishing, as long as it satisfies the syntax in the definition. In other words, a distinguisher is not necessarily a classifier, even though that's normally what one should have in mind.

To formalize the notion of how good a distinguisher is, we'll define a *validation test* next. This is exactly what we would normally think of as cross-validation, or any other leave-some-out kind of validation typically usually used in machine learning:

Definition A *validation test* is an experiment involving two random variables P_0 and P_1, and a distinghuisher $\mathcal{D}$. A uniform random bit $Z \sim \mathrm{B}(0.5)$ is sampled, and then, accordingly, $X \sim P_Z$ is sampled. The distinguisher is given X and yields z. If $z = Z$ the *value of the test*, denoted $V(P_0, P_1, \mathcal{D})$, is 1, otherwise it is 0.

This is a test to see if $\mathcal{D}$ can guess whether it was shown a sample from P_0 or P_1. Note that the sample X must be a random variable *from the perspective of* $\mathcal{D}$. That is, we cannot re-use an X that had been used to construct D, as might be done during "training". Generally, when the random variables and distinguisher are implied, I'll write $V(P_0, P_1, \mathcal{D})$ simply as V.

We are now ready to define "algorithmic distinctness", which we'll just call *distinctness*, denote it by θ:

Definition: Two random variables P and Q have *distinctness* θ if the *best* distinguisher $\mathcal{D}$ achieves accuracy $\frac{1+\theta}{2}$ in a validation test with P and Q. That is,

$$\theta = 2 \sup_{\mathcal{D}} \mathrm{E}\{\mathrm{V}(P, Q, \mathcal{D})\} - 1 \tag{4}$$

I'll denote the *best* distinguisher for the distributions P and Q by the symbol D_{PQ} and will refer to it as the *canonical distinguisher for P and Q*. Formally:

$$\mathcal{D}_{PQ} = \arg\max_{\mathcal{D}} \mathrm{E}\{\mathrm{V}(P, Q, \mathcal{D})\} \tag{5}$$

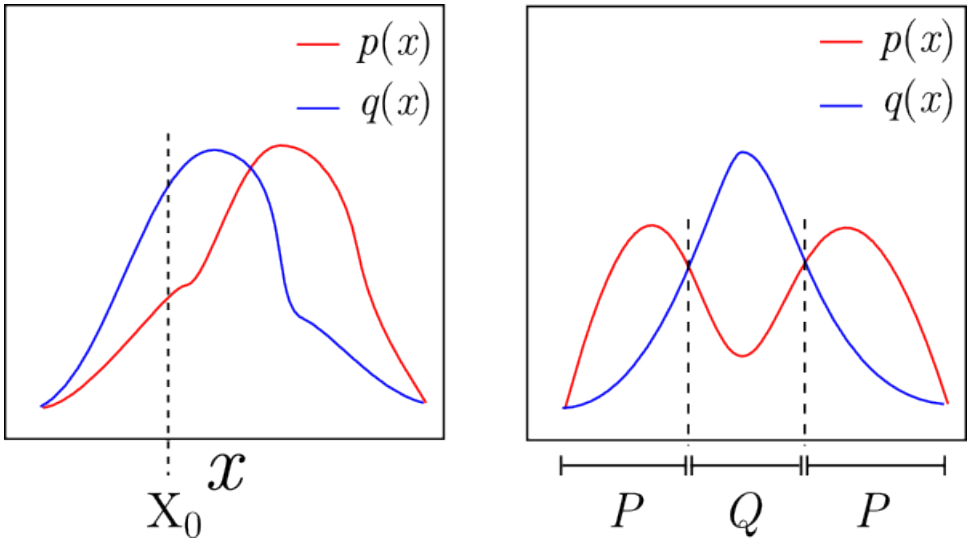
0.6 Connections

Nearly all measures of divergence are interconvertible, being merely transformations of one another (The reason to have many is that each has its own interpretation). The fact that TV, JSD, and θ are interconvertable should not come as a surprise.

Each serves an important role in putting forward a simple and practical way to talk about the stochasticity of HPUs. We will see that θ provides a solid justification for approaching the measurement of divergence using classifiers. JSD on the other hand, has an interesting interpretation which can be likened to the "leakage" of side-stream information into HPU outputs. TV provides a common currency of "difference", serving as a link between θ and the more orthodox divergence measures.

0.7 Claim: $\theta = TV$

Let us return to Fig. 0.3. In general, given two random variables P and Q, we don't know what their distributions, $p(x)$ and $q(x)$ look like, and trying to determine TV involves repeatedly sampling in order to build up a picture of $p(x)$ and $q(x)$. But let us assume, for the sake of argument, that we have exact



analytical expressions for $p(x)$ and $q(x)$, and they are as given in the first panel of Fig. 0.3. In that case, how well will the *best* classifier, $\mathcal{D}_{PQ}$ perform?

Suppose that $\mathcal{D}_{PQ}$ receives an X_0 as shown in Fig. 0.7. What should be its guess? The only information that $\mathcal{D}_{PQ}$ has to work with is the value of X_0, and given that $p(X_0) > q(X_0)$, it must guess that $X_0 \sim P$. This is true in general so D_{PQ} necesarily embeds an exact model of regions where $p(x) > q(x)$, as depicted in the second panel of Fig. 0.7. Of course, $\mathcal{D}_{PQ}$ does not need to know the actual value of $p(x)$ or $q(x)$ to produce the best possible guesses; it only needs to know the regions for which one is strictly higher than the other.

Since the best classifier, $\mathcal{D}_{PQ}$ must follow this policy of guessing, we can easily calculate its accuracy in validation. Since we take tests half of the time from P and half of the time from Q, X comes from the mixture distribution, $X \sim m(x) = \frac{1}{2}p(x) + \frac{1}{2}q(x)$.

Let's begin the proof that $\theta = TV$.

Proof: Given some X, the probability that $\mathcal{D}_{PV}$ guesses correctly is

$$\Pr\{V = 1|X = x\} = \frac{\max(q(x), p(x))}{p(x) + q(x)} \tag{6}$$

Note that we can write $\max(q(x), p(x))$ in a nicer way:

$$\max(q(x), p(x)) = \frac{q(x) + p(x)}{2} + \frac{|p(x) - q(x)|}{2} \tag{7}$$

So,

$$\Pr\{V = 1|X = x\} = \frac{1}{p(x) + q(x)} \left(\frac{q(x) + p(x)}{2} + \frac{|p(x) - q(x)|}{2} \right) \tag{8}$$

$$= \frac{1}{2} + \frac{|p(x) - q(x)|}{2(p(x) + q(x))} \tag{9}$$

Now, to get the overall probability of success, we multiply the conditional probability of success given $X = x$ by the probability that $X = x$, over the whole support of X. Remember, X has the mixture

distribution during the validation test, $m(x) = \frac{p(x)+q(x)}{2}$:

$$\Pr\{V = 1\} = \sum_{x \in \mathcal{X}} \Pr\{V = 1 | X = x\} \Pr\{X = x\} \tag{10}$$

$$= \sum_{x \in \mathcal{X}} \left(\frac{1}{2} + \frac{|p(x) - q(x)|}{2(p(x) + q(x))} \right) \left(\frac{p(x) + q(x)}{2} \right) \tag{11}$$

$$= \frac{1}{2} \sum_{x \in \mathcal{X}} m(x) + \frac{1}{2} \left(\frac{1}{2} \sum_{x \in \mathcal{X}} |p(x) - q(x)| \right) \tag{12}$$

$$= \frac{1}{2}(1) + \frac{1}{2} TV \tag{13}$$

$$= \frac{1 + TV}{2} \tag{14}$$

Now, since V is a Bernoulli random variable,

$$\mathrm{E}\{V\} = \Pr\{V\} \tag{15}$$

$$\frac{1 + \theta}{2} = \frac{1 + TV}{2} \tag{16}$$

$$\implies \theta = TV \tag{17}$$

■

So we have that θ and TV are really the same thing. What is not clear, yet at lest, is how that will help us.

We'll get to that! But first, let's finish making connections.

0.8 Link to JSD

We'll begin by calculating JSD from the probability densities, and derive a relationship to θ. While doing so, the mathematics will reveal an interesting interpretation of JSD with regard to HPU processing.

Here, I'll leave off the argument for the probability densities, so I'll write p, q, and m to mean $p(x)$, $q(x)$, and $m(x)$, meanwhile let's agree that the sums implicitly range over all $x \in \mathcal{X}$. Beginning from the definition in terms of the Kullback-Leibler divergence:

$$JSD = \frac{1}{2} \sum p \lg \left(\frac{p}{m} \right) + \frac{1}{2} \sum q \lg \left(\frac{q}{m} \right) \tag{18}$$

$$= \frac{1}{2} \sum \left[-(p + q) \lg m + p \lg p + q \lg q \right] \tag{19}$$

$$= - \sum \left[m \lg m \right] + \frac{1}{2} \sum \left[p \lg p + q \lg q \right] \tag{20}$$

$$= H(X) - H(X|Z) \tag{21}$$

$$= I(X; Z) \tag{22}$$

In the second from last line, we've substituted in the entropy of X and the conditional entropy of X given Z based on their definitions. Recall that Z is the random bit that we draw during the validation test in order to decide which distribution to pick X from. When we take the entropy of X and subtract from it the remaining entropy in X after we know Z, the difference is equal to how much we learn about X when Z is revealed, i.e. the mutual information of X and Z, $I(X; Z)$.

So, JSD is the amount of information we gain about X if we are told what Z is.

Interpreting JSD as the information we gain about a mixture variable X from knowing whether it was drawn from P or Q is a well known interpretation. But in our case it is particularly interesting because

it actually matches the validation experiment we set up. Here, JSD measures the amount of information available to distinguishers during the validation test.

Mutual information is symmetric, i.e. $I(X;Z) = I(Z;X)$, which allows us to obtain:

$$JSD = H(Z) - H(Z|X) \tag{23}$$

Since $Z \sim \mathrm{B}(0.5)$, $H(Z) = 1$. Meanwhile, $H(Z|X)$ is remaining uncertainty about Z once X is revealed, which is precicely the uncertainty of the canonical distinguisher $\mathcal{D}_{PQ}$, on average, after it is provided with X, and this gives us the connection with θ:

$$\begin{aligned} H(Z|X) &= H\{V(P,Q,\mathcal{D}_{PQ})\} && (24)\\ &= H\left\{\mathrm{B}\left(\frac{1+\theta}{2}\right)\right\} && (25)\\ &= \frac{1+\theta}{2}\lg\left(\frac{1+\theta}{2}\right) + \frac{1-\theta}{2}\lg\left(\frac{1-\theta}{2}\right) && (26)\\ &= 1 - \frac{1}{2}\left[(1+\theta)\lg(1+\theta) + (1-\theta)\lg(1-\theta)\right] && (27) \end{aligned}$$

And so JSD is given by θ in the following way:

$$\begin{aligned} JSD &= H(Z) - H(Z|X) && (28)\\ &= 1 - \left(1 - \frac{1}{2}\left[(1+\theta)\lg(1+\theta) + (1-\theta)\lg(1-\theta)\right]\right) && (29)\\ &= \frac{1}{2}\left[(1+\theta)\lg(1+\theta) + (1-\theta)\lg(1-\theta)\right] && (30) \end{aligned}$$

0.9 Interpretations

The initial motivation for θ is to measure how different, or *distinct*, one HPU population is from another, in terms of the population of outputs that they produce while completing a task. This applies when one population has been primed one way, and another population has been primed another way, and we wish to decide how much "bias" is introduced by the priming treatement. In most cases it would probably make sense to think about one treatment as the "correct" or intended treatment, and the other as the "biasing" treatment, a treatment which one wishes to avoid subjecting HPUs to since it will affect the output in some unexpected or undesireable way. Their isn't any asymetry built into the definitions though.

Pursuing this idea, another way to interpret θ is as the maximum *distortion* suffered by a binary decision made on the basis of HPU outputs, where the distortion is brought about by allowing HPUs to be primed to be like Q when they should have been primed to be like P.

To make this clear, suppose that we make some decision, D, on the basis of the output of an HPU, whose outcome we encode as either 0 or 1. What is the effect, on the decision D of using an HPU primed to have output distributed like Q, vs one primed to have output distributed like P, under some task $\mathcal{T}$?

It would seem that the worst-case scenario would be that HPUs distributed like P would give us output that would always lead us to decide 0, while HPUs distributed like Q would give us output that would always lead us to decide 1.

There are two things to notice about this possibility though. First, this is a strange "decision", since, provided we prime the HPUs properly, they will give output that always leads us to the same decision, and hence, we probably don't actually need to consult them. The second thing to notice, is that D would be functioning as a perfect distinguisher. Generally, this will be impossible, unless $\theta = 1$. No distinguisher can function better than the canonical distinguisher, so the probability that the decision get

"flipped" due to the fact that the HPU was primed can never be greater than θ. Therefore, θ bounds the amount of bias introduced into yes/no decisions, made on the basis of an HPU output, due to priming.

At this point, its important to note that this is only true for binary decisions. I'm not sure yet what happens for multy-way decisions.

But sticking to binary decisions for now, we can also see another interpretation for in JSD. The JSD between the desired prime and the undesired prime is equal to the information that "leaks through" into the HPU's output due to that prime. Again, this is with respect to a binary decision. Binary decisions include such things as conditionals in a larger a HPU algorithm or whether or not to flag this image is innappropriate.

I have seen a generalization for JSD to n distributions. It should be possible to figure out how much information leaks from priming into a multi-way decision, but I'm not sure how hard that will be.

0.10 Classifiers calculate the tightest lower bounds for θ / TV / JSD

We have not yet discussed how one might go about actually measuring θ. Since it was defined in terms of the *best possible classifier*, it might seem like the connection with classifiers is merely theoretical. Although it yields interesting interpretations and connections, so the reasonning might go, it is not a measurement tool.

This couldn't be further from the truth. I will now argue that classifiers are the best technology we have for measuring θ. To make my point, I'd like to step away from classifiers and look at the alternative.

Looking back to Fig. 0.3, we saw that TV (a.k.a. θ) is defined based on areas between the probability densities of two distributions. Why not sample outputs from P and Q to build an empirical representation of their distributions, then calculate θ from that?

In principle, and in many cases in practice, that will work, but there are major difficulties that are not obvious. To illustrate this, let's begin with the example where we have asked HPUs to label an image with 5 labels.

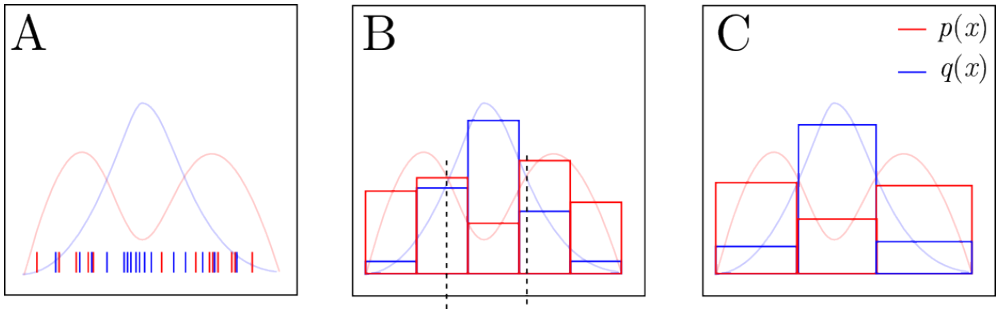
In principle, every single ordered 5-tuple is a unique x that lives somewhere in the distribution of X. This is a large space, in the sense that we probably will never see the same x twice during the experiment. But in practice, of course, it is debatable as to whether the occurence of a word in position 1 is different from its occurence in position 5. So, we might choose to collapse the support of X (which is $\mathcal{X}$) down into a multinomial with $n = 5$.

This is probably a fortuitous decision. Of course, this is not how we actually think of words, because if one X had `cat` and `dog` in it, we would suspect that it is a *different* `dog` than that in an X containing `hot` and `mustard`.

There is nothing terribly surprising here, and we know that one way to try to get at these relationships is with SVD, and I guess LDA? The point is to realize that we are compressing $\mathcal{X}$, lumping various x's together that, in, the raw output, were distinct. This is not necessarily benign, but it is often unavoidable. The question is, how does this affect the calculation of θ.

To make visualization easier as we answer this question, let's use a low dimensional example. In Fig. 0.10 we collapse the raw samples shown in panel A into the histograms shown in B and C, based on a different binning policies. In fact, even panel A depicts a histogram, its just that the rectangles are so skinny, and all the bins either have zero or one element, since we're imagining a sparse distribution.

Now panel B has finer bins than panel C, so, if we calculate all the differences between blue and red rectangle areas, it should give a better estimate of θ right? Wrong, In fact. It isn't the fineness of bins, but the placement of their endpoints that matters.



If a bin spans the intersection of the two distributions, which happens twice in pannel B (see the black dashed lines), then that bin's contribution to θ will be reduced. This is because part of it covers a region of the distribution where $p(x) > q(x)$, and part of it covers an area where $q(x) > p(x)$, and these partially cancel out. In C the bins line up perfectly with the *true* cross-over points of the distributions, it is an optimal binning policy, as is any policy that doesn't make bins span cross-over points. Using that binning will enable the best estimate of θ as is possible through the observation of samples.

I hope these claims are intuitive enough that the reader can convince themselves that they are true. A proof would be in order, and I think I should do it, but it will take me away from the point that I am trying to make.

The difficulty is that we don't know ahead of time what the cross-over points are, because we are trying to measure the distributions themselves! Naturally, to calculate θ based on the differences in empirical sample frequencies at particular locations, we'll need to infer the regions in which $p(x) > q(x)$ (and vice versa), and line up our bins with the boundaries of these regions.

But surely this is algorithmically tractible. Aren't there algorithms which 'learn' the best boundaries for bins as they sample?

What we need is an algorithm that partitions the support, demarcating the areas for which a P is more likely to be sampled, from those for which a Q is more likely. Once we do that, we just sample each random variable N times, and keep track of the difference in the number of occurences in each area.

Does this sound familiar? Read that paragraph carefully, then read the following 'translation':

What we need is a classifier, that divides up the set of possible documents or vectors ($\mathcal{X}$), into the contiguous areas in which one distribution's samples are more likely to fall than the other's. Then we can just run some validation test, to see how well the classifier performs.

One difference, with using the classifier is that, if we use a validation test, then some subset of the data are used only to train the classifier, while the rest was used to test it in order to locate and establish confidence intervals for θ.

If it seems clear that using the binning approach in B would produce an *underestimate*, now consider trying to calculate θ directly from the raw samples in A.

If we have a method to determine θ which is better than any classifier we have, then we can turn it into a new classifier that is more accurate than any classifier we have. Simply use the method, and extract from it the bins it uses to calculate differences. Observe which bins have more Ps or more Qs, and as soon as a new example is given, assign it the majority class inside its bin. Based on the equivalence of θ and TV, the classifiers accuracy must be equal to the calculated TV.