To get to the idea of shared information between lineup and play style, we must define “information”. We’ll use the information theoretic definition of information called “entropy”. Entropy in information theory, like entropy in thermodynamics, measures the spread of a probability mass function or a probability density function or in other words, how random the process is that follows the function. The definition is below:

To illustrate the utility of the equation, imagine a biased coin that has heads and tails. The probability that you flip heads is 100% and 0% for tails. In the above equation one would get . In this context, the second term is defined as 0 or . The resulting entropy is 0. The function has not spread or randomness. Sometimes, the term is used as the average information content of a distribution. is large when lower probabilities exist in the distribution and these low probability outcomes are considered “surprising” which can carry more information. Then entropy would be the average of all the events which will be larger if there are more surprising (low probability) events. There is an analog for continuous distributions, although it cannot be compared one-to-one with discrete entropy. For example, the formula can result in negative information content.

Now that there is a framework for getting the information content of a distribution, what does it mean for two distributions to share information? One could imagine that for two random variables, knowledge of the outcome of one variable might contain some information about the other. For example, there could be two random variables, one where it is sunny or cloudy and another where it’s raining or not. It could be that it only rains when it's cloudly but it doesn’t always rain when its cloudy. Then if you know it’s sunny, you also know it’s not raining. In this scenario, information about one random variable gives nearly complete information about the other. Information is shared. The sharing of information would look something like:

This says that random process Y has some information but knowledge about X reduces the information in Y. The amount of that reduction is the shared information. In information theory, this is called mutual information an is more formally defined as

This shows that if two random variables are independent (), Then the mutual information is 0. This can also be written in the continuous case with integrals

The continuous case is particularly important as the player position metrics are continuous, yet the lineups are discrete. It’s possible to mix these variable types and still arrive at a mutual information that looks like equation XX. See derivation below where X is the lineup random variable and Y is the player position vectors as a continuous random variable.

This derivation is particularly useful for this problem. The entropy of the continuous variable is challenging to acquire, for reasons discussed in the next section however it is possible. The conditional entropy, , is then possible if we use the formulation from the step before . An entropy for each class in x is created and then a weighted average of the entropies is taken.

**Estimating continuous entropy**

Although the continuous or “differential” entropy equation is simple, evaluating it is difficult given that the probability density function is not known – one can only see the observations sampled from . There are many ways to estimate the differential entropy. The general methods are a histogram technique, kernel density estimation of the probability density function, or a nearest neighbor approach. The first and third methods are investigated.

A straight forward method to dealing with the continuous case is to remove the continuity and discretize the function. The integral can be broken down into a sum of N intervals where properties about the interval are approximated as constant. The formulation is below

Then for each interval , the probability density is constant and thus is constant and can be moved outside the integral resulting in:

can be interpreted as the probability mass in interval k and approximated by taking the width of the interval and multiplying by the constant probability density in that interval. In other words:

So then subbing into equation XX, we retrieve the histogram estimator:

Notice that discretizing the differential entropy did not result in the true discrete entropy. Differential entropy is not quite the same as discrete entropy and to account for this difference, the width is included which can be interpreted as a bias of sorts.

The positives to this method is that on its face, it’s simple. There is also some control for the level of resolution. Given the number of samples, it may be useful to have higher resolution to properly capture the distribution, or with fewer data points, reduce the resolution as to not let some bins dominate the estimate by chance.

The second method is to use the distance from one point to its kth nearest neighbor to determine the density around the point itself. Intuitively, if the samples are relatively close on average, there must be less entropy in the distribution generating those samples.

If one observes equation XX, it is clear that this is an expectation of . An estimate of the expectation would be to forego the integral and directly calculate

For all data points in the sample. This is still a challenge as we do not know the form of the probability density function – we do not know . As we’ll see later, it may be possible to estimate the logarithm of the probability density function at . The estimator will take on the form

We’ll start with the approximation that the probability mass around a point is equal to a sufficiently small volume multiplied by a constant probability density around that point – more formally:

Where is the distance from point to its kth nearest neighbor

is the probability mass of the volume around point to its kth nearest neighbor

is the volume of a hypersphere in d-dimensional space with radius 1

Then if the logarithm of both sides is taken, it’s clear that an estimate of the logarithm of the probability density at point is related to the logarithm of the probability mass. Focus is turned to getting a representation for the logarithm of the probability mass.

Consider a probability density function that is the probability density that is the distance to the kth nearest neighbor of point i. Then is the probability mass that one point, point i’s kth nearest neighbor, is contained in the range . Then exactly k-1 points are at distances less than the kth nearest neighbor and N-k-1, the rest, are at distances farther then the kth nearest neighbor. This is a trinomial distribution which will take on the form

The probability of success, exactly one point in the region is the incremental probability density multiplied by the width of that region, the probability of being inside that region was defined in eqn XX and the probability of being outside that region is the remaining probability mass. The factorials in front have been simplified using the definition of “choose” from combinatorics.

Given that we now have an expression for the probability density of the distance to the kth nearest neighbor, we can now generate the average of the logarithm of the probability mass around a point to its kth nearest neighbor.

Substituting equation XX and moving constants out from the integral we obtain:

Notice the limits of integration have changed. After simplification, the variable being integrated over is no longer but the probability mass which is only supported from 0 to 1. Simplifying the integral requires knowledge of the Beta Function which is defined as

And also equals

Where is the gamma function. Then taking the derivative of the Beta function

or

Then using the definition of the digamma function:

And rearranging

Then

Substituting into the partial derivative of the Beta function

And simplifying

Then the integral in equation XX simplifies

Plugging in this definition for our case we retrieve

Substituting into equation XX, we return our estimate for the logarithm of the probability density at point i.

All that’s left is to take the negative average over :

The benefits of this method is that the one assumption being made is about the constant probability density in the k nearest neighbor sphere, but with small k and sufficiently large dataset, this error can be small. It can also be more computationally efficient than binning on high dimensional data.

It should be pointed out that the errors that arise from assuming constant probability density in the volume that contains the k nearest neighbors will add up the more discrete classes there are. There is a technique to mitigate this error when calculating mutual information. As pointed out earlier, mutual information is:

When converting to our estimators, we get

Firstly, notice that has been moved to inside the average for the marginal entropy rather than held constant for all points. This is because it will no longer be held constant for all points in the marginal entropy. K’s in the conditional entropy can be different from those in marginal entropy. Then K’s can be selected such that the distance to the kth nearest neighbor in the conditional distribution is the same as the distance to a different kth nearest neighbor in the marginal distribution for all points. Which will cancel those distances and remove some of the error from the constant density assumption. With careful selection of k’s, equation xx simplifies to

For example, take some point in the conditional distribution and its nearest neighbor. That same nearest neighbor point is chosen as the nearest neighbor in the marginal entropy. Given there are other points from other classes that will be in the marginal, will be greater than or equal to .

It should be noted that there are still errors from calculating an average probability mass around a point rather than the exact probability mass even without the uniform density assumption. There is also error simply from sampling, however these errors are significantly less than those that come from the uniform density assumption.

**Description of results**