Divariance: Connecting Frechet, KL Divergence and Variance for sufficient feature selection

Looking more deeply into a metric used for sufficient feature selection in previous article, showing derivation, intuition on use and show how it relates to existing measures of distance/divergence over elliptical curves. Ultimately show it Root Dirrelation Divergence can be a proxy for KL Divergence when attempting to identify important class features and differentiated data.

from Steve Morrell https://www.stevemorrellart.com/latest-work) with permission of the artist

KL Divergence and Frechet distance are used all over machine learning: feature selection, , especially sufficient/efficient selection where targeting specific classes or questions. Data drift, model drift and regime change in time series and projections. Computer vision models and LLMs it is used to make sure trained and fine-tuned model distributions adhere closely to ground truth or base model distributions.

In a previous case (https://towardsdatascience.com/pairwise-cross-variance-classification/), I have used it to determine which dimensions of a CLIP model are sufficient to differentiate between two classes of images/text. In sufficient/effective dimension reduction, you are NOT attempting to reduce the features of the model across all the targets in order to reduce the input size in general. You are reducing the feature input size ONLY when asking specific questions: is this a kitten or a puppy? A totally different set of dimensions may be needed if asking: is this a car or a jet? For many models, its easy to tell the difference between a shoe and a dress, but requires some fine tuning to differentiate between a dress and a skirt. By identifying only the effective dimensions and getting rid of the noise of the ineffective dimensions, you can increase the difference in the cosine similarity between potential targets.

Now I am digging into metric I used to identify effective dimensions. It belongs to the family of frechet distances (used in training GANS amongst other things) but I approached it from the perspective of comparing

variances directly. Ultimately, it evolves into a normalized metric that approximates average KL Divergence for normali-like distributions.

My objectives for this article are to show:

- both KLD and Frechet distance can be calculated in terms of divariance for elliptical distributions.
- Root Dirrrelation Divergence (RDD) follows from divariance normalized over the product of variances.
- how the measures differ as means and variances of synthetic data diverge.
- RDD adheres more closely with Empirical KL Divergence than formulaic KL divergence for two normal distributions.
- advantages of RDD over KL Divergence in terms of speed, simplicity and smoothness.

A data scientist reading this article will:

- learn about mathematical connections between variance, frechet and KLD
- See opportunities for cross pollination of ideas between different metrics. Understand where you could substitute formulaic values for samples for speed and interpretability.

There are other approaches to this problem, but I find this one simple to calculate and update, and effective in quantifying the salient features in a model for a particular question. I've provided links to some research on the topics but will not get too deeply into them in this article. Check (https://www.stat.uchicago.edu/~lekheng/work/probdist.pdf) for an overview and a demonstration of how divergences and distances relate.

Intuition on metrics

KL Divergence (KLD) and Frechet Distance (FD) are methods for calculating how different two distributions are from one another - both in mean and variance. Specifically, I'm looking at elliptical distributions that are defined by mean and variance, so that is typically enough. Variance of course measures the dispersion of observations within a distribution. Divariance is a distance metric that will be on the same scale as variance (and converges to variance as the distributions converge). You can find the github with code here.

I will repeatedly refer to these arrays:
X_1 & X_2 are random variables from normal distributions P & Q:
$X_1 \sim p(x)x; P \sim \mathcal{N}(\mu_1, \sigma_1^2)$ with n samples
$X_2 \sim q(x)x; Q \sim \mathcal{N}(\mu_2, \sigma_2^2)$ with m samples

I will generate related normal distributions with the following code:

```
def generate_distributions(mean1, mean2, var1, var2, corr, size=1000):
    """
    Generate two distributions with specified means, variances, and covariance.

    Parameters:
        mean1 (float): Mean of the first distribution.
        mean2 (float): Mean of the second distribution.
        var1 (float): Variance of the first distribution.
        var2 (float): Variance of the second distribution.
        #cov (float): Covariance between the two distributions.
```

```
        corr (float): correlation between the two distributions
        size (int): Number of samples to generate.

    Returns:
        x (np.ndarray): Samples from the first distribution.
        y (np.ndarray): Samples from the second distribution.
    """
    # calculate covar
    cov = corr*np.sqrt(var1)*np.sqrt(var2)

    # Create the covariance matrix
    cov_matrix = np.array([[var1, cov],
                           [cov, var2]])
    # Mean vector
    mean_vector = np.array([mean1, mean2])
    # Generate samples
    samples = np.random.multivariate_normal(mean_vector, cov_matrix, size)
    x, y = samples[:, 0], samples[:, 1]
    return x, y

# Example usage:
x, y = generate_distributions(mean1=0, mean2=0, var1=1, var2=2, corr=1, size=1000)
```

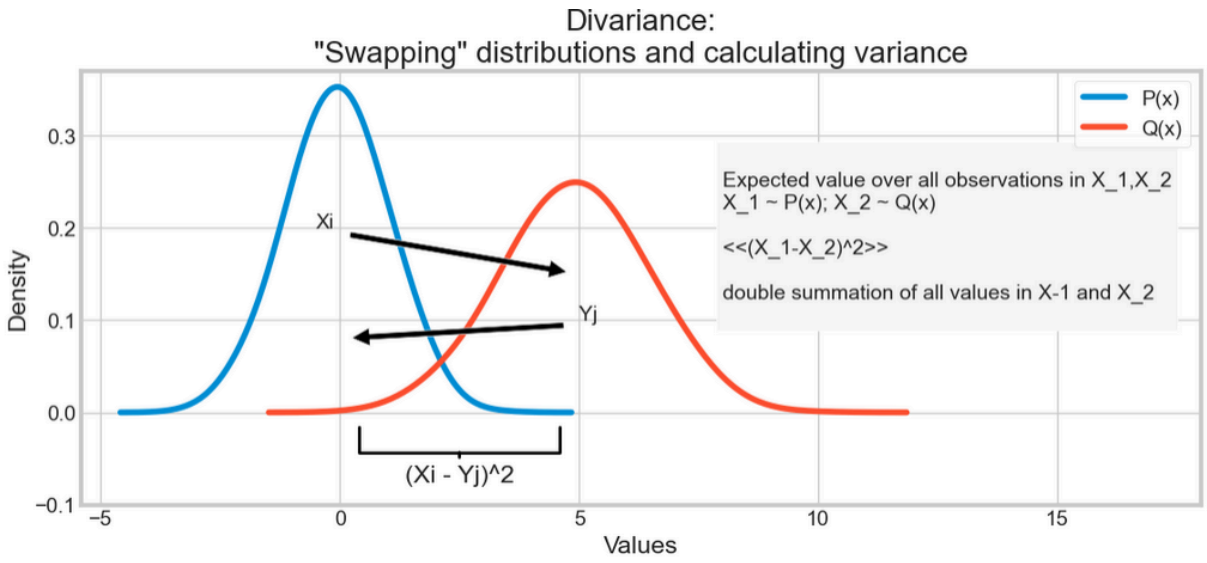
Divariance

Divariance is a distance metric that will be on the same scale as variance. It is always greater or equal to the product of the two standard deviations (similarly covariance is always less than or equal to variance). Divariance converges to variance as the distributions converge.

If you have two distributions, $X = X_1$ and $Y = X_2$, then the divariance ς^2_{XY} ("var-sigma squared of XY") is:

$$\varsigma_{XY_2} = \sum_{i}^{n} \sum_{j}^{m} \frac{(x_i - y_j)^2}{2mn}$$

The idea is to calculate the incremental variance for systematically swapping the class assignments - or how much more distant are the members of the other class vs the actual. You calculate this by swapping the distribution for an observation from X to Y, then calculate the variance contribution.



Instead of calculating the difference vs the mean $\sum(x_i - \mu_1)^2$ as in variance, you calculate the difference vs each member of the other distribution $\sum\sum(x_i - y_j)^2$. You then calculate a row of differences for each observation in X_1 and end up with an n x m matrix of squared differences.

$$\varsigma^2_{XY} = \left\langle \begin{pmatrix} (x_1 - y_1)^2 & (x_1 - y_2)^2 & \cdots \\ (x_2 - y_1)^2 & (x_2 - y_2)^2 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} \right\rangle$$

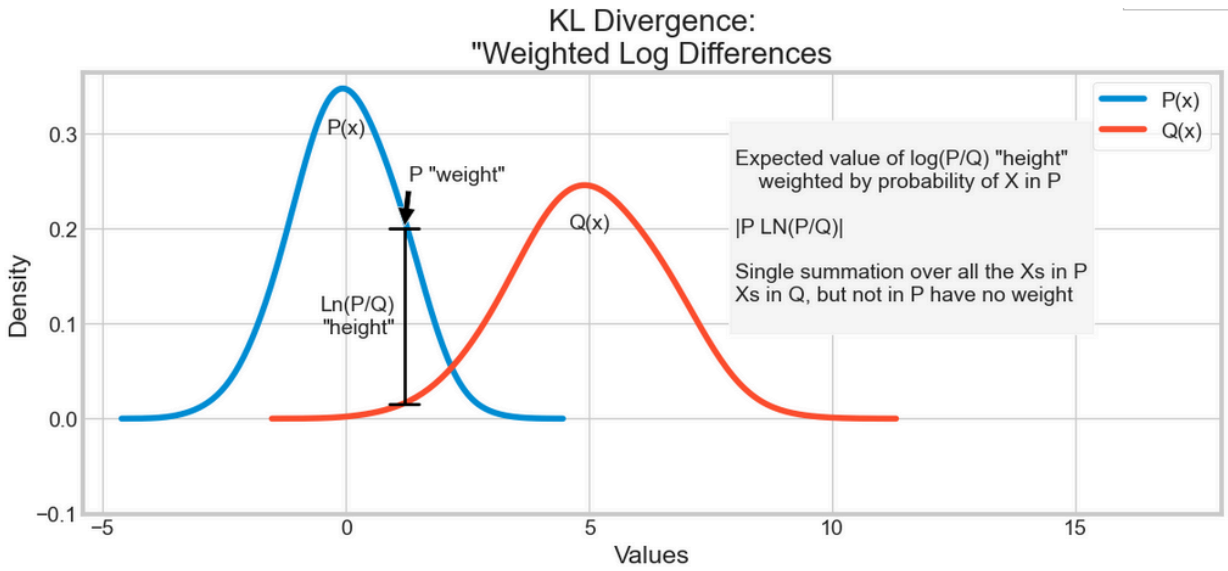
If you then take an average of each row over the m observations in X_2, you have a the per member variance contribution of swapping each observation of X_1 to X_2. Similarly, if you average the columns over n observations in X_1, you have a the per member variance contribution of swapping each observation of X_2 to X_1. If you divide the entire matrix by 2mn, you get the expected value of the squared differences, which intuitively is the variance after swapping every observation.

KL Divergence (KLD):

Description: One of the most common methods in data science for comparing distributions. Take the distributions above and use X_1 is your ground truth distribution and X2 is the distribution you are optimizing. The get the KL divergence of Q from P, sum the log of the probability of x in P divided by the probability of x in Q, weighted by the probability of x in P.

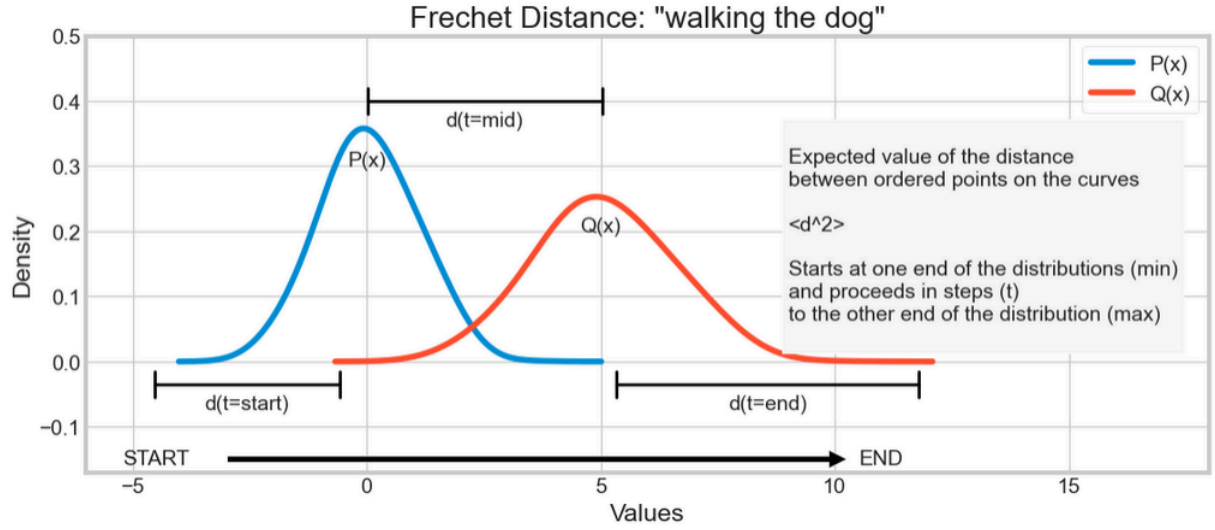
$KL_{P|Q} = \int P(x) LN(\frac{P(x)}{Q(x)})$ and conversely $KL_{Q|P} = \int Q(x) LN(\frac{Q(x)}{P(x)})$

The choice of the ground truth distribution matters: $KL_{P|Q} <> KL_{Q|P}$. The metric will be different based on weightings and you can visualize that if you optimize using $KL_{P|Q}$, then the red distribution will move towards the blue (vs converging in the middle).



Frechet Distance (FD):

Description:While KLD measures the “vertical” distance between distributions at a given value of x, FD measures the distance in both the “horizontal” and “vertical” between the “start” of the two curves on the left. Its been characterized as the expected value of distance between a person(P) and dog(Q) when starting at one end of distributions and “walking the dog” to the other side.



You start at the leftmost end of each curve at time t_start and note the distance between the points on the curve, then progress to t+1 all the way to the end at t_end, accumulating the distance. Thus the Frechet Distance is the expected value for the accumulated distances:
$d^2(P,Q) =< (x_t - y_t)^2 >$. The intuition is a little odd, but this metric is central to training GANS and in other multidimensional classification/optimization functions. Below I’ll introduce the formula for FD applied to normal curves, which is more tractable.

Definitions and Derivations:

As noted above:
X_1 & X_2 are random variables from normal distributions P & Q:

$X_1 \sim p(x)x; P \sim \mathcal{N}(\mu_1, \sigma_1^2)$ with n samples
$X_2 \sim q(x)x; Q \sim \mathcal{N}(\mu_2, \sigma_2^2)$ with m samples

Derivation of divariance:

Similar to variance, but for two distributions
Divariance formula (var sigma):

$$\varsigma_{ij} = \sum_i^n \sum_j^m \frac{(x_i - x_j)^2}{2mn} \qquad \text{(eq. d1)}$$

$$= \frac{1}{2mn} \sum_i^n \sum_j^m [x_i^2 - 2x_i x_j + x_j^2]$$

$$= \frac{1}{2mn} \Big[\underbrace{\sum_i^n \sum_j^m x_i^2}_{\text{does not vary w/j}} \underbrace{- 2 \sum_i^n \sum_j^m x_i x_j}_{\dots} + \underbrace{\sum_i^n \sum_j^m x_i^2}_{\text{does not vary w/i}} \Big]$$

$$= \frac{1}{2mn} \Big[m \sum_i^n x_i^2 \quad -2 \sum_i^n \sum_j^m x_i x_j \quad + n \sum_j^m x_j^2 \Big]$$

divide all by 2mn:

$$= \frac{\cancel{m} \sum_i^n x_i^2}{2\cancel{m}n} \quad - \frac{\cancel{2} \sum_i^n \sum_j^m x_i x_j}{\cancel{2}mn} \quad + \frac{\cancel{n} \sum_j^m x_j^2}{2m\cancel{n}}$$

$$= \underbrace{\tfrac{1}{2} \sum_i^n x_i^2 / n}_{1/2 \text{ mean of } X_i^2} - \underbrace{\sum_i^n x_i}_{\text{mean of } X_i} \underbrace{\sum_j^m x_j}_{\text{mean of } X_j} + \underbrace{\tfrac{1}{2} \sum_j^m x_j^2 / m}_{1/2 \text{ mean of } X_j^2}$$

$$= \frac{1}{2} \bar{x_i^2} \qquad - \bar{x_i}\bar{x_j} \qquad + \frac{1}{2} \bar{x_i^2}$$

$$= \frac{\bar{x_i^2} + \bar{x_j^2}}{2} \qquad - \qquad \bar{x_i}\bar{x_j} \qquad \text{(d2)}$$

$$= \text{mean of squares} \qquad - \qquad \text{square of means}$$

The code for calculating divariance goes from computing a square matrix and taking the mean:

```
np.power((arr_a[:] - arr_b[:,None]),2).mean(axis=(0,1))/2
# or (faster) -
difference = (arr_a[:] - arr_b[:,None])
cv = np.einsum('ij...,ij...->...',difference,difference)/(len(arr_a)*len(arr_b)*2)
```

to calculating means over x_1, x_2, x_1^2, x_2^2 and combining:

```
def divariance(arr_a, arr_b):
    """dvr: divariance    variance of swapped members of two distributions
```

```
    I subtract one from denomanator to match covariance... only meaninful for small
sample sizes"""
    a_mean, b_mean = arr_a.sum(axis=-1)/(arr_a.shape[-1]-1),
arr_b.sum(axis=-1)/(arr_b.shape[-1]-1)#arr_a.mean(axis=-1), arr_b.mean(axis=-1)
    a_sqr_mean, b_sqr_mean = np.power(arr_a,2).sum(axis=-1)/(arr_a.shape[-1]-1) ,
np.power(arr_b,2).sum(axis=-1)/(arr_b.shape[-1]-1)#np.power(arr_a,2).mean(axis=-1) ,
np.power(arr_b,2).mean(axis=-1)
    dvr = (a_sqr_mean+b_sqr_mean)/2 - a_mean*b_mean
    return dvr
```

To see the relationship between divariance and variance, recall the formula for variance (σ^2) as a partition of expected values:

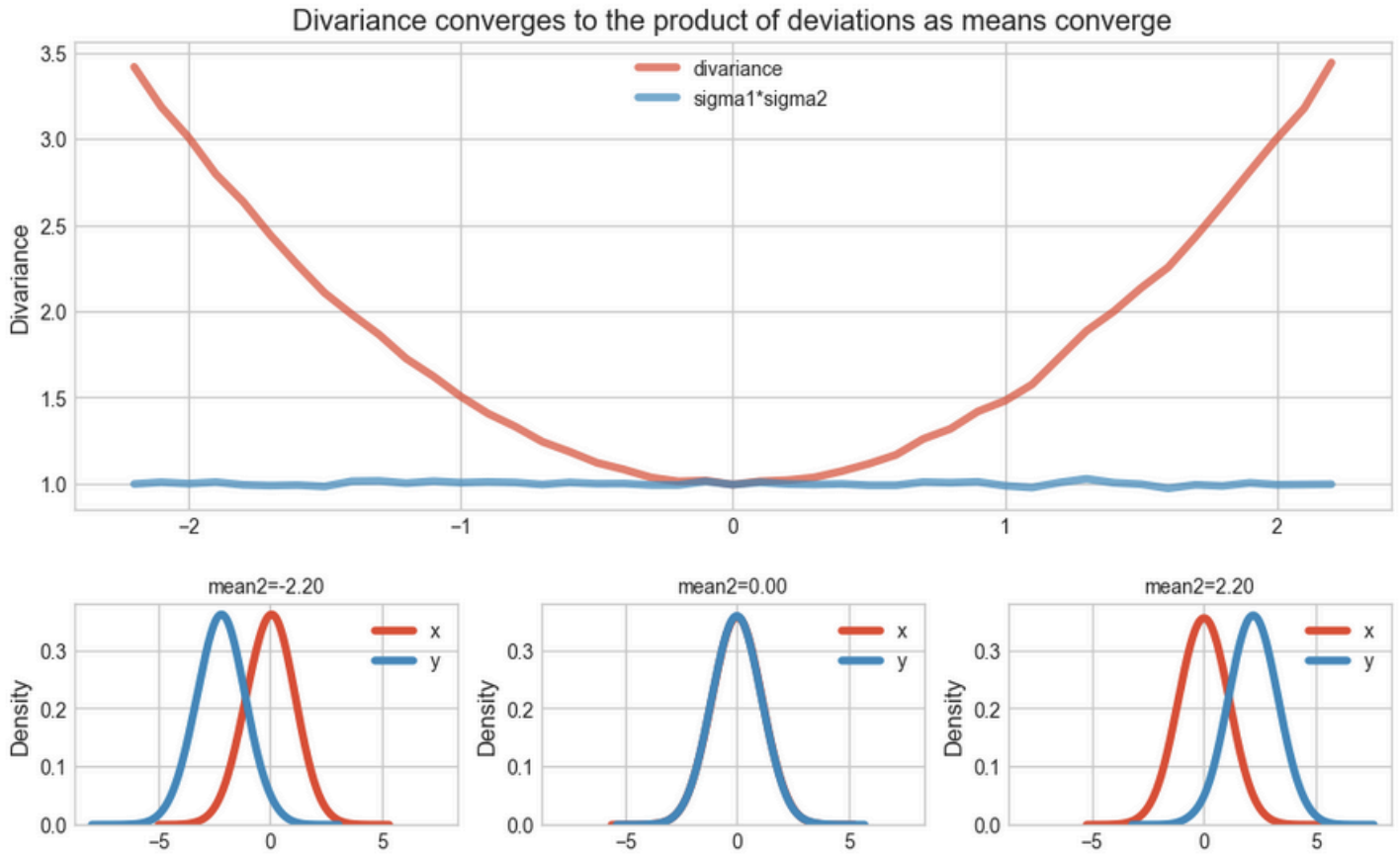
$\sigma_i^2 = \bar{x_i^2} - \bar{x_i}^2$ (you can see the derivation here) (v1)

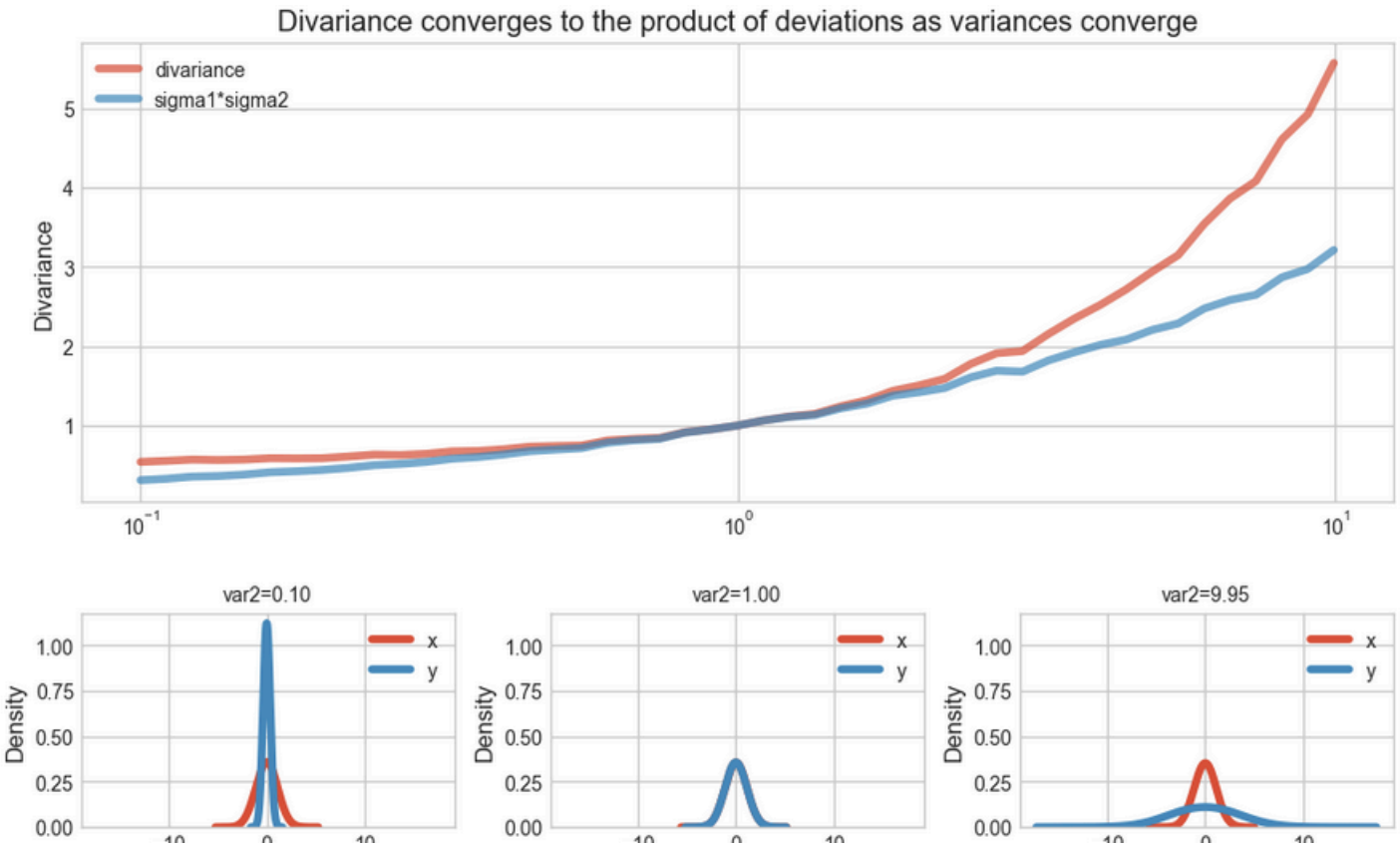
Substitute $x_j \to x_i$ into (d2):

$= \frac{\bar{x_i^2} + \bar{x_i^2}}{2} - \bar{x_i}\bar{x_i}$

$= \bar{x_i^2} - \bar{x_i}^2 = \sigma_i^2$ (v1)

So as the distributions converge, divariance converges to variance.





Simplifying:

Putting divariance in terms of μ and σ:

Re-arrange (d2) and substitute μ for mean of x:

$$\varsigma_{12}^2 = \frac{1}{2}\left[\bar{X_1^2} - 2\mu_1\mu_2 + \bar{X_2^2}\right] \qquad \text{(d3)}$$

Note that variance as a partition of expected values (v1) implies:

$\bar{x_i^2} = \sigma_i^2 \quad + \quad \bar{x_i}^2$

Now, substitute in to (d3):

$$\frac{1}{2}\left[\sigma_1^2 + \mu_1^2 - 2\mu_1\mu_2 + \sigma_2^2 + \mu_2^2\right]$$

Group by like terms:

$$\frac{1}{2}\left[\sigma_1^2 + \sigma_2^2 + \mu_1^2 - 2\mu_1\mu_2 + \mu_2^2\right]$$

combine the square and you have divariance defined neatly in terms of μ and σ:

$$\varsigma_{12}^2 = \frac{1}{2}\left[\sigma_1^2 + \sigma_2^2 + (\mu_1 - \mu_2)^2\right] \qquad \text{(d4: divar(mu,sigma))}$$

```
def divariance(mean1,mean2,var1, var2,corr=0,size=0):
    '''divariance from moments'''
    return (np.power(mean1-mean2,2)+var1+var2)/2
```

Standardized over the product of $\sigma_1\sigma_2$: Dirrelation

If we put divariance ((d3) over $\sigma_1\sigma_2$ analogously to correlation you get:

$$\varrho = \frac{\varsigma_{12}^2}{\sigma_1\sigma_2} = \frac{(\mu 1 - \mu 2)^2 + \sigma_1^2 + \sigma_2^2}{2\sigma_1\sigma 2} \qquad \text{(r1) dirrelation}$$

```
def dirrelation(mean1,mean2,var1, var2,corr=0,size=0):
    return divariance_m(mean1,mean2,var1, var2)/(np.sqrt(var1*var2))
```

Which I've been called "dirrelation" until I find it referenced otherwise: while "co"-relation measures how pairs of observations move together relative to standard deviation, "di"-relation indicates how the combined variances of the distributions diverge relative to standard deviations. For labelling, I used "var-rho" (ϱ) to represent dirrelation, similarly to "rho" (ρ) represents correlation.

Dirrelation converges to 1

Note that the product of sigmas converge to variance as σ_1 converges to $\sigma_2\,\sigma_2$

$\sigma 2 -> \sigma 1; \sigma 1\sigma 2 -> \sigma_1^2$... ie variance.

So for (r1) we showed both the numerator and denominator will converge to variance as the distributions become more similar, meaning the ratio will converge to 1.

Dirrelation >= 1

The numerator will always be greater than or equal to the denominator, so dirrelation will always be >= 1. Divariance is greater or equal to the product of standard deviations:

$$\frac{1}{2}\left[\sigma_i^2 + \sigma_j^2 \quad + \quad (\mu_i - \mu_j)^2\right] >= \sigma_j\sigma_i$$

move terms to the left:

$$\frac{1}{2}\left[\sigma_i^2 + \sigma_j^2 \quad + \quad (\mu_i - \mu_j)^2\right] - \sigma_j\sigma_i >= 0$$

bring standard deviations inside bracket

$$\frac{1}{2}\left[\sigma_i^2 + \sigma_j^2 \quad - 2\sigma_j\sigma_i + \quad (\mu_i - \mu_j)^2\right] >= 0$$

combine the square:

$$\frac{1}{2}\left[(\sigma_i - \sigma_j)^2 \quad + \quad (\mu_i - \mu_j)^2\right] >= 0$$

Since both terms are squared, it will never be less than 0, but will approach 0 as mean and standard deviation converge. Also, note that inside the brackets is the equation for frechet distance when pulled from two normals or "any two distributions from the family of real, elliptically contoured distributions" defined by mean and variance, and familiar density function (I'll call that a 2-moment distribution for short). From the 1983 paper:

$$d^2(F, G) = (\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

(f1)

```
def frechet_distance_norms(mean1,mean2,var1, var2,corr=0,size=0):
    return (np.power(mean1-mean2,2)+np.power(var1-var2,2))
```

So in demonstrating that divariance will always be greater than the product of sigmas, we arrive at Frechet Distance (FD) for 2-moment distributions always being greater than 0. FD has has been a foundational distance metric for multi-feature image training, amongst other things.
Break the sigmas term apart and put (f1) in terms of divariance and our familiar P and Q distributions:

$d^2(P,Q) = 2(\varsigma_{12}^2 - \sigma_1\sigma_2)$ `(f2) frechet_distance(divariance)

and conversely:

$\varsigma_{12}^2 = \frac{1}{2}d^2(P,Q) + \sigma_1\sigma_2$ (d3) divariance(frechet_distance)

Convexity

Its second derivative with respect to both $\mu \text{ and } \sigma$ will be greater than 0, so it will be convex for any changes in the difference of means or variance.
The second derivative of (r1) with respect to the difference of means (m):

$$\frac{\partial^2}{\partial m^2}\left(\frac{m^2 + (s_1 - s_2)^2}{2\,s_1\,s_2}\right) = \frac{1}{s_1\,s_2}$$

The second derivative of (r1) with respect to the standard deviation (s_1):

$$\frac{\partial^2}{\partial {s_1}^2}\left(\frac{m^2 + (s_1 - s_2)^2}{2\,s_1\,s_2}\right) = \frac{m^2 + s_2^2}{s_1^3\,s_2}$$

Since standard deviation and m are always non-negative, this will be positive (or undefined), indicating convexity.

Frechet Divergence: a zero based metric

Now subtract 1 to base it around 0, which will put it in common divergence terms, although not explicitly scaled by probabilities.

$$\frac{\varsigma_{12}^2}{\sigma_1\sigma_2} - 1 = \frac{2\varsigma_{12}^2}{2\sigma_1\sigma_2} - 1$$

Get a common denominator by multiplying 1 by $2\sigma_1\sigma_2$

$$= \frac{(\mu1 - \mu2)^2 + \sigma_1^2 + \sigma_2^2}{2\sigma_1\sigma_2} - \frac{2\sigma_1\sigma_2}{2\sigma_1\sigma_2}$$
$$= \frac{(\mu1 - \mu2)^2 + \sigma_1^2 + \sigma_2^2 - 2\sigma_1\sigma_2} {2\sigma_1\sigma_2}$$
Combine the square of the sigma terms:
$$= \frac{(\mu1 - \mu2)^2 + (\sigma_1 + \sigma_2)^2} {2\sigma_1\sigma_2}$$
The numerator is the Frechet for a 2-moment distribution (eq f1), which yields a “frechet divergence”:
$$\frac{d^2(P,Q)}{2\sigma_1\sigma_2} = \frac{\varsigma_{12}^2}{\sigma_1\sigma_2} -1 = \varrho - 1$$
(r2)

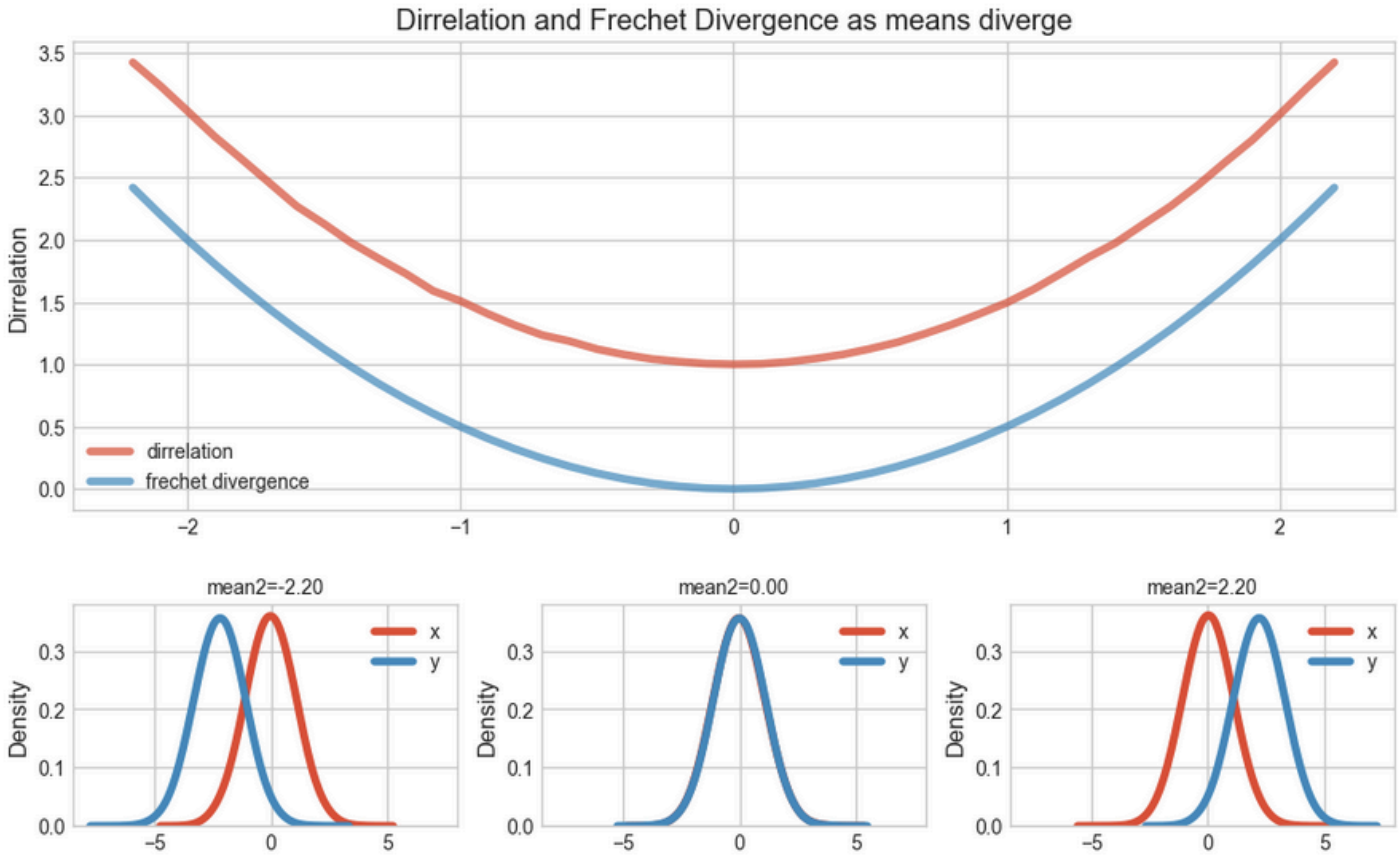
This gives us a version of the Frechet divergence that is, as I’ll show empirically is:

- convex

- centered and minimized around 0
- symmetrical around differences in μ
- log-symmentrical around differences in σ^2.

The above works as a divergence metric when dealing with two 2-moment distributions when there is not a preferential prior or posterior… you are testing how variance would marginally increase if you cumulatively swapped each sample into the other distribution. As you'll see below, it doesn't track KL Divergence when both means and variances diverge.

```
def frechet_divergence(mean1,mean2,var1, var2,corr=0,size=0):
    return frechet_distance_norms(mean1,mean2,var1, var2,corr=0,size=0) /
(2*np.sqrt(var1*var2))
```



Relationship to KL Divergence.

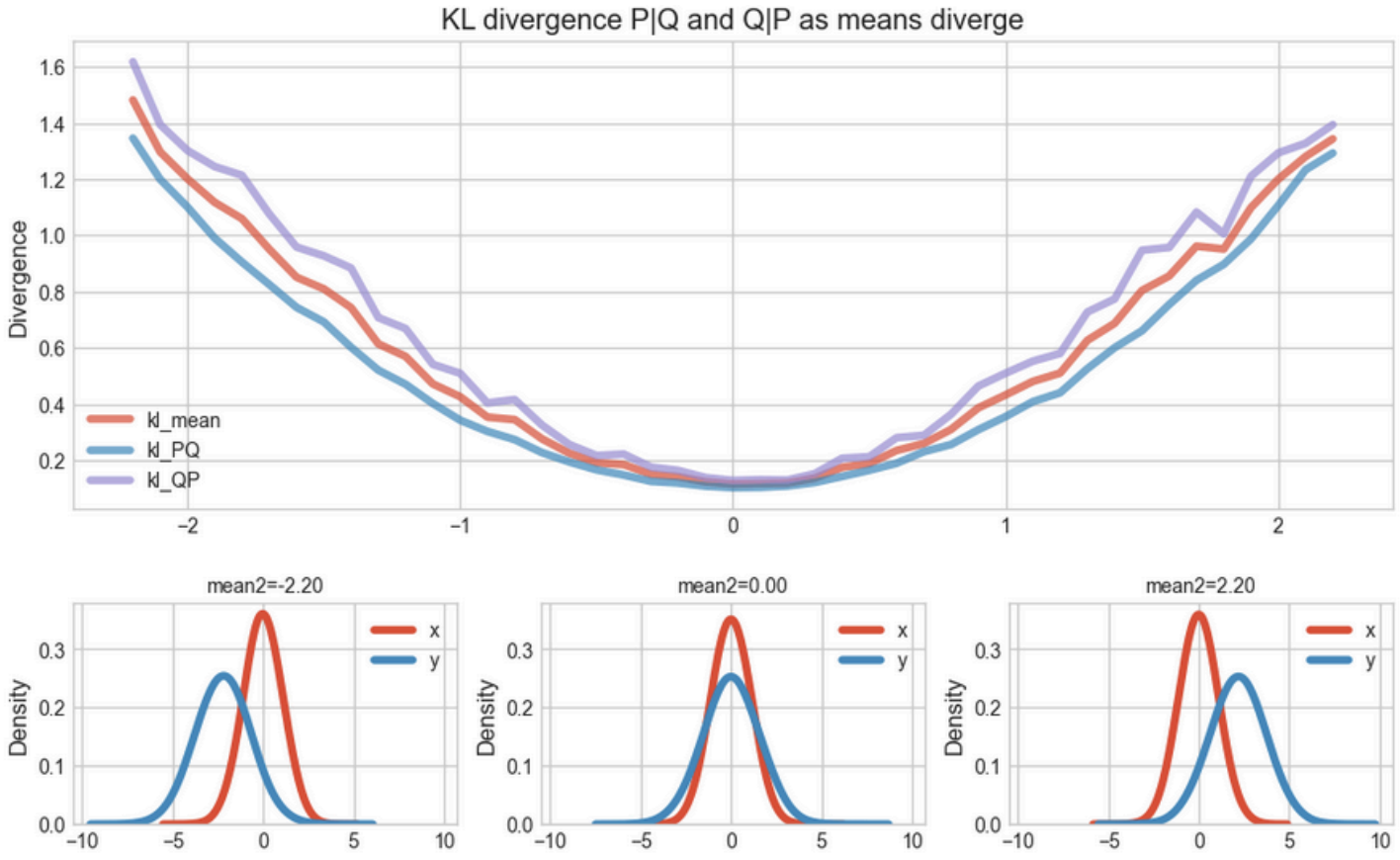
Divergences are distinct from distances because they are not symmetrical. Its nicely explained here. Its units are probability density. Distances have units related to the underlying arrays.

It was noted that KL divergence for two sample normal distributions is non-symmetrical. However, if we take an arithmetic average, it become symmetrical in some instances. Below, I generate various normal distributions with differing target means and variances, keeping the first distribution with (μ, σ^2) = (0,1).

Here, we use the following defaults (note var2 = 2):

distr_default = {'mean1':0, 'mean2':0, 'var1':1, 'var2':2, 'corr':0.75, 'size':10000}

then vary mean2 between (-2.2 and 2.2):



To calculate the actual KL divergence, I have to turn the arrays into probability distributions, so I generate bins between the global max and min of the two distributions. Then I count the proportion of the total array within each bin. The result will be very close to a normal distribution. Then plug the empirical distributions for p(x) and q(x) into the KL divergence formula:

```
def dist_from_range(arr,drange):
    return
((arr[:]>drange[:-1,np.newaxis])&(arr[:]<=drange[1:,np.newaxis])).sum(axis=1)/len(arr)

from scipy.special import kl_div

def kl_div_from_arrays(x,y,bins=100):
    min_val = min(np.min(x),np.min(y))-0.01
    max_val = max(np.max(x),np.max(y))
    drange = np.linspace(min_val, max_val,bins)
    x_pd = dist_from_range(x,drange)
    y_pd = dist_from_range(y,drange)
    return sum(np.nan_to_num(kl_div(x_pd,y_pd),posinf=0))
```

I also want to see what a closed form theoretical value for KL divergence of two normals would be. Part of identifying a closed form metric is that I do not need to build bins and counts when calculating. If the metric is calculated in terms of mean and variance, that is easy to calculate, update and optimize.
Using the proof, you can see the formula for KLD of two normals.

$$KL_{P|Q} = \frac{1}{2}\big[\frac{(\mu_1 - \mu_2)}{\sigma_2^2} + \frac{\sigma_1^2}{\sigma_2^2} - LN(\frac{\sigma_1^2}{\sigma_2^2}) - 1$$

Similarly, the KLD of P relative to Q:

$$KL_{Q|P} = \frac{1}{2}\big[\frac{(\mu_1 - \mu_2)}{\sigma_2^2} + \frac{\sigma_1^2}{\sigma_2^2} - LN(\frac{\sigma_1^2}{\sigma_2^2}) - 1$$

```
def kl_normal(mean1,mean2,var1, var2,corr=0,size=0):
    return 0.5*((var1+np.power(mean1-mean2,2))/var2 - np.log(var1/var2) - 1)
```

Joining terms and distributing the fraction, you get:

$$KL_{P|Q} = \frac{(\mu_1 - \mu_2) + \sigma_1^2}{2\sigma_2^2} - \frac{1}{2}LN(\frac{\sigma_1^2}{\sigma_2^2}) - \frac{1}{2} \qquad \text{(k1)}$$

note that from eq (d2):

$$\varsigma_{12}^2 - \frac{\sigma_2^2}{2} = \frac{\sigma_1^2 + (\mu_1 - \mu 2)^2}{2}$$

So (k1) becomes:

$$KL_{P|Q} = \frac{\varsigma_{12}^2 - \frac{\sigma_2^2}{2}}{\sigma_2^2} - \frac{1}{2}LN(\frac{\sigma_1^2}{\sigma_2^2}) - \frac{1}{2}$$

$$KL_{P|Q} = \frac{\varsigma_{12}^2}{\sigma_2^2} - \frac{1}{2} - \frac{1}{2}LN(\frac{\sigma_1^2}{\sigma_2^2}) - \frac{1}{2}$$

$$KL_{P|Q} = \frac{\varsigma_{12}^2}{\sigma_2^2} - \frac{1}{2}LN(\frac{\sigma_1^2}{\sigma_2^2}) - 1 \qquad \text{(k2)}$$

In (k2) you can see that KL divergence for a two 2-moment distributions can be defined in terms of divariance and variances. This also shows that KL_{P|Q} can be defined in terms of a frechet distance via eq (d4).

For the purposes of demonstration, I take an average of KL_{P|Q} and KL_{Q|P} to create a symmetrical divergence that will be comparable to the Frechet Divergence (r2).

$$\frac{1}{2}\big[KL_{P|Q} + KL_{Q|P}\big] = \frac{1}{2}\big[\frac{\varsigma_{12}^2}{\sigma_2^2} - \frac{1}{2}LN(\frac{\sigma_1^2}{\sigma_2^2}) - 1 \quad + \quad \frac{\varsigma_{12}^2}{\sigma_1^2} - \frac{1}{2}LN(\frac{\sigma_2^2}{\sigma_1^2}) - 1\big]$$

$$= \frac{1}{2}\big[\frac{\varsigma_{12}^2}{\sigma_2^2} + \frac{\varsigma_{12}^2}{\sigma_1^2} - 2 - \frac{1}{2}\big(LN(\frac{\sigma_1^2}{\sigma_2^2}) + LN(\frac{\sigma_2^2}{\sigma_1^2})\big)\big]$$

Note:

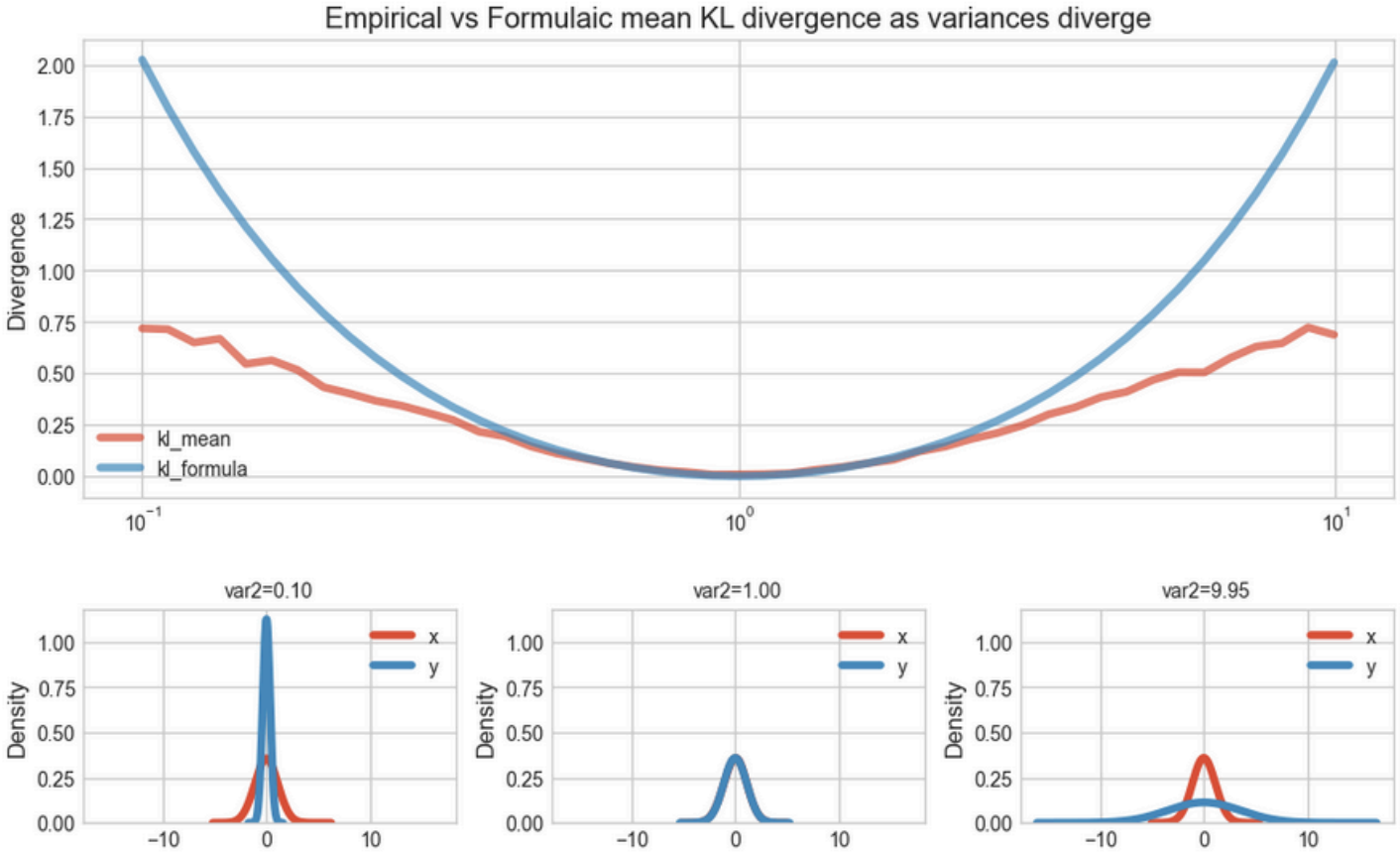
$$LN(\frac{\sigma_1^2}{\sigma_2^2}) + LN(\frac{\sigma_2^2}{\sigma_1^2}) = LN(\frac{\sigma_1^2}{\sigma_2^2}\frac{\sigma_2^2}{\sigma_1^2}) = LN(1) = 0$$

So the above equation simplifies to an expected value (arithmatic average):

$$< KL_{P|Q}, KL_{Q|P} >= \frac{\varsigma_{12}^2}{2\sigma_2^2} + \frac{\varsigma_{12}^2}{2\sigma_1^2} - 1 \qquad \text{(k4)}$$

For the purposes of identifying the gain a particular feature adds in differentiating between two classes, a symmetrical measure makes sense - there is no preferential class.

```
def kl_mean_from_formula(mean1,mean2,var1, var2,corr=0,size=0):
return 0.5*( (var1+np.power(mean1-mean2,2))/(2*var2) +
(var2+np.power(mean1-mean2,2))/(2*var1) -1)
```



Note that the formulaic version of KLD suffers from extreme sensitivity to the variance of the alternate distribution (Q). If it gets close to the 0, this formula explodes upwards. Practically, if Q(x) ~ 0, KLD is undefined, so this isn't seen. This ends up reflected in the mean KLD and is a good reason to to depend on the formulaic version. Taking a geometric mean mitigates the problem, but doesn't solve it. This might be a good reason to use a Frechet Divariance to estimate sample KLD when variances can get close to 0 - it is less sensitive to a the alternate variance because the two variances are multiplied together in the denominator. For optimization purposes, you may want the convexity of the formula when means and variances diverge substantially… the calculated KLD tends to flatten out at some point and may not yield a consistent direction for gradients. Thats a bit beyond this article.

Root dirrelation - dampening the extremes:

I note above how the formulaic version of KL divergence can deviate substantially from observed KL divergence when the means and variances are quite different. The same occurs to a lesser extent for the Frechet Divergence (r2) because it multiplies variance in the denomenator, which can get very small very quickly. You will see this in the empirical examples below.

To account for this, i take a square root of the dirrelation formula (r1) to put it into “standard deviation” space (the linear space that standard deviation and mean exist in, vs the “squared space” that variance exists in). “root var-rho”:

$$\sqrt{\varrho} = \sqrt{\frac{\varsigma_{12}^2}{\sigma_1 \sigma_2}} = \frac{\varsigma_{12}}{\sqrt{\sigma_1 \sigma_2}} \quad \text{(r3)}$$

I have found that the following “root dirrelation divergence” (RDD) matches more closely with empirical mean KLD.

$$RDD = 2 * (\sqrt{\varrho} - 1) \quad \text{(r4)}$$

The dampening function $z(c^{1/z} - 1)$ is convergent to some fraction of c for integers of z greater than 1; the choice of z=2 is simple and adequate. The dampened divergence (RDD) has the same characteristics of as the Frechet divergence in terms of convexity, zero basis and symmetry.

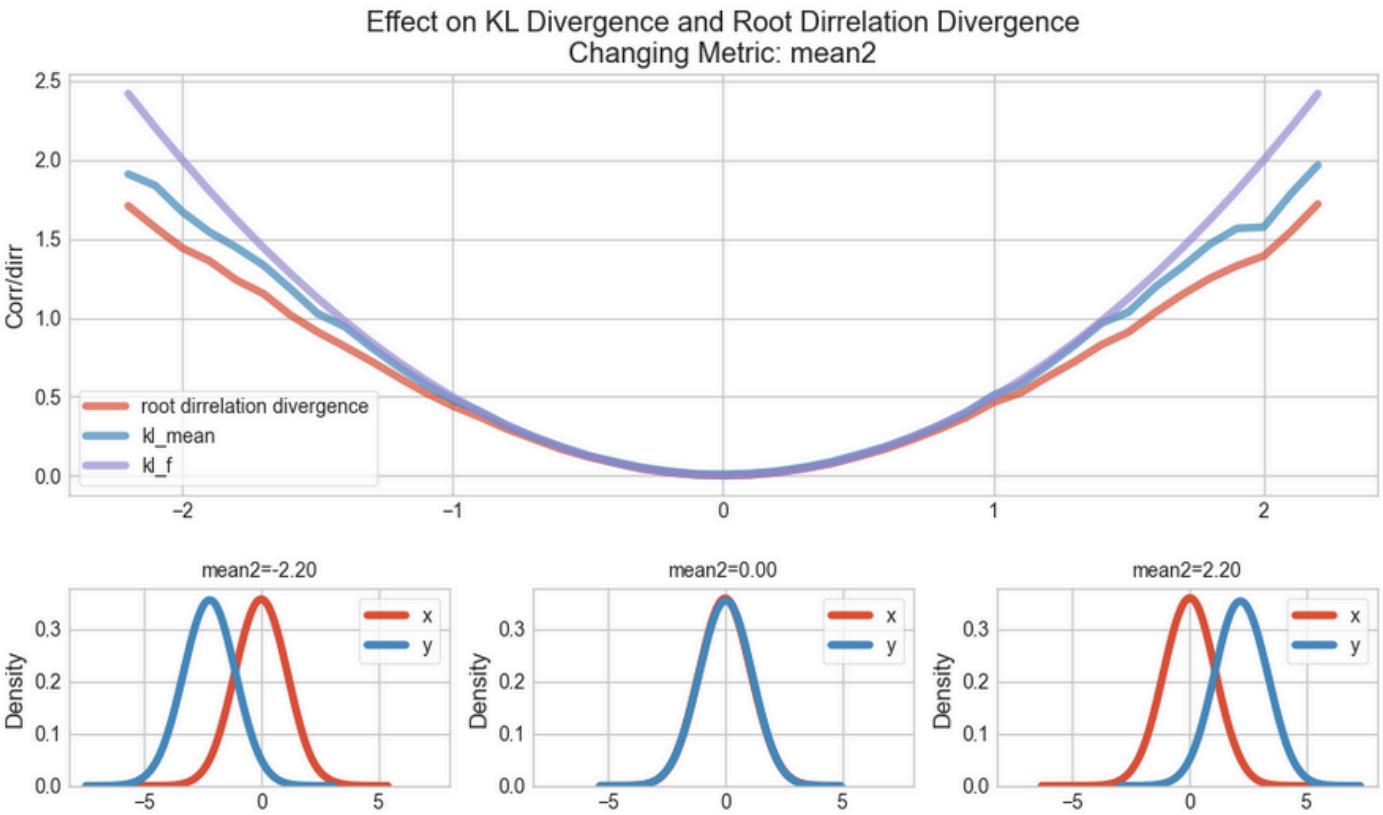
```
def divergence_dampening(x,y=2):
    """ x: divergence value
        y: dampening metric.  Integers will converge to a fixed value quickly so 2 is
recommended.  Pass y=f(x) if dynamic dampening is desired
    """
    return np.round(y * (np.power(x,1/(y))-1),3)

def RDD(mean1,mean2,var1, var2,corr=0,size=0):
    return divergence_dampening(dirrelation(mean1,mean2,var1, var2), 2)
```

Sample Distribution Examples

Below are examples of how the divergence metrics change as the distributions diverge by changing either the mean or the variance of the second distribution.
Two distributions are generated based on the following default values:

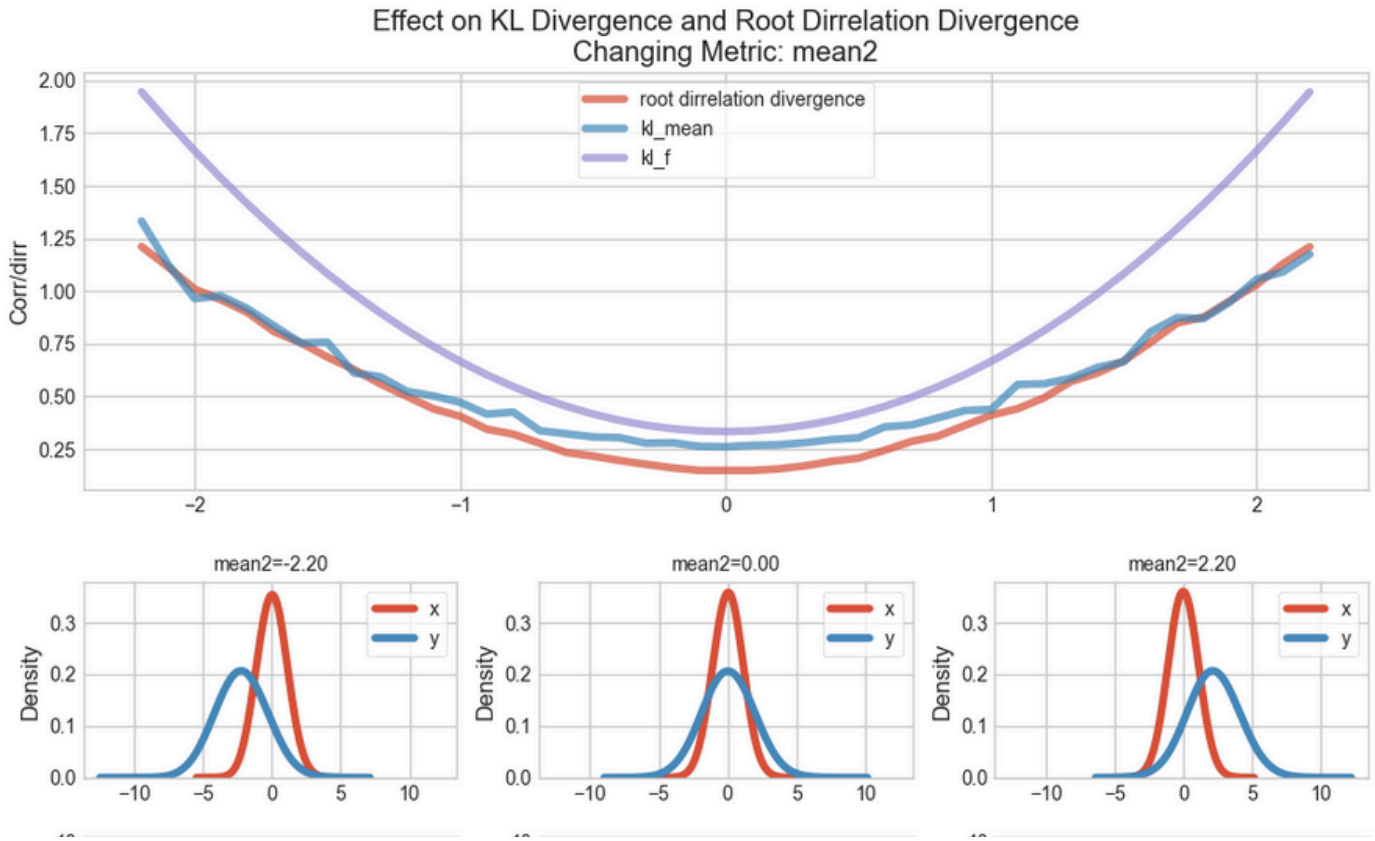
distr_default = {'mean1':0, 'mean2':0, 'var1':1, 'var2':1, 'corr':0.75, 'size':10000}

The 1) root dirrelation divergence, 2) empirical mean KL divergence (kl_mean) and the 3) KL divergence based on the formula for normals (kl_formula) are captured in the top chart. The Second row of charts show how the density distributions shifts as the chosen “Change Metric” varies. Correlation is set to 0.75, but it doesn’t affect any of these statistics.



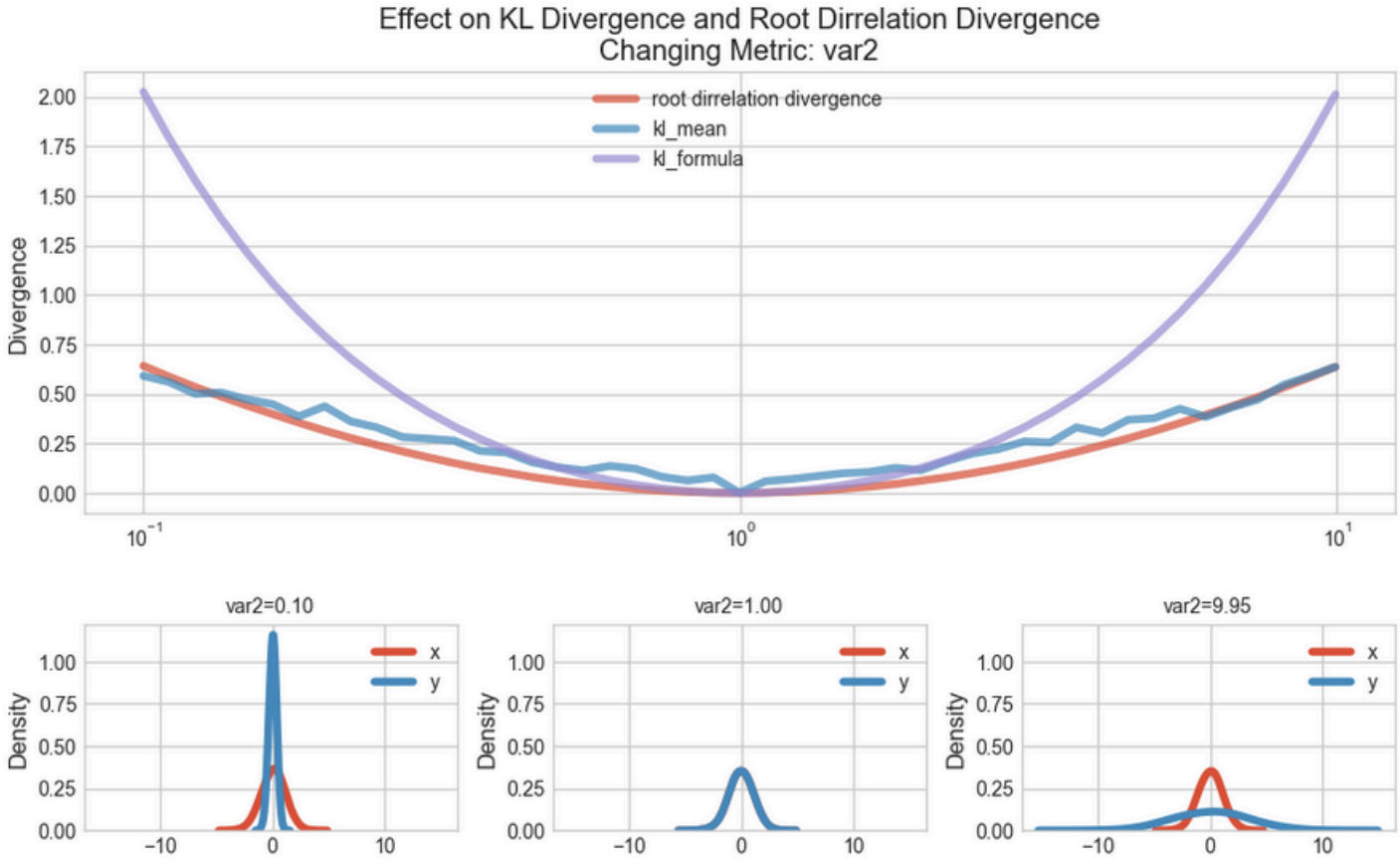
Here we hold the variances equal and change the mean of the y-distribution between -2.2 and 2.2. Both the RDD (red) and KL_f (purple) track the empirical KL divergence relatively well. The gradient of the KL_f is starting to ramp, but not terribly.

Now, let's shift the variance upwards for the y-distribution from var2=1 to var2=3 so the starting parameters look like this:
distr_default = {'mean1':0, 'mean2':0, 'var1':1, 'var2':3, 'corr':0.75, 'size':10000}



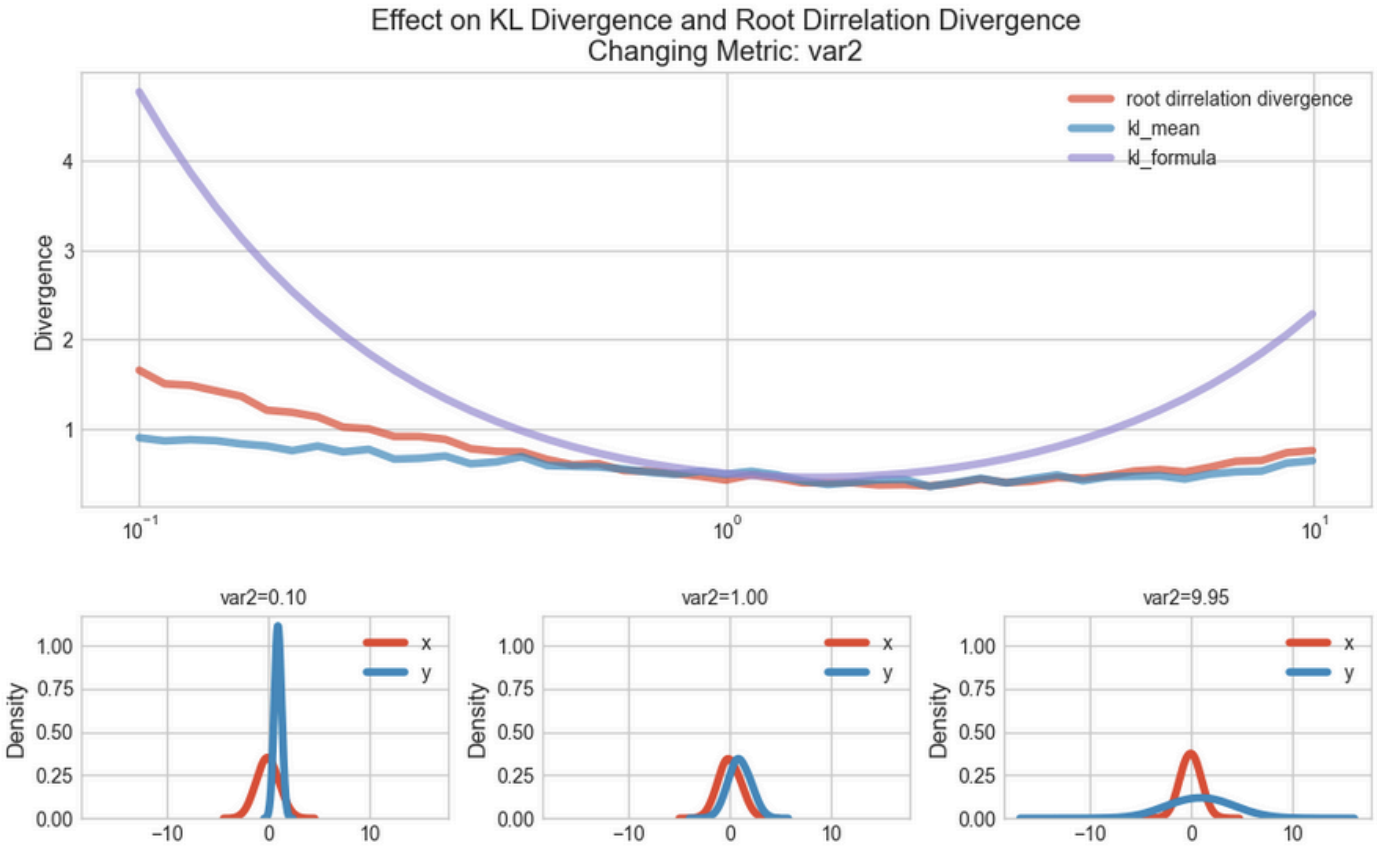
You can see that the KL_f starts to ramp pretty intensely at the extremes, diverging from the empirical KL_mean. The RDD underestimates KL_Mean when the means are similar (in the middle), but stays convex and “catches up” as the means diverge. Also note the “bumpiness” of the empirical KL mean, which can lead to some problems when optimizing.

Now we hold the difference of means constant (at zero) and see the effect of shifts of variance. We change the scale to a log basis to maintain symmetry and show var2 in a range between 0 and 10:



The metrics are all log-symmetrical. RDD underestimates KL_mean when the variance are similar, but tracks well otherwise. KL_formula diverges substantially at extremes as the relative variances causes the denominators to approach zero and the gradient explodes.

Finally, we set mean2=1 and show the same range for var2:



RDD eventually diverges as var2 approaches 0 (on the left) but otherwise closely tracks the empirical KL_mean. KL_formula is over-convex.

Advantages of RDD over empirical mean KL Divergence

I've shown that RDD closely matches mean KL Divergence for varying mean and variances when applied to sample data. The reasons to use it when selecting features or differentiating model distributions are:

1. Speed
2. Simplicity
3. Smoothness

Measuring KL distribution on two raw arrays requires constructing an intermediate probability distribution with shared x-axis values, then constructing the differences piecemeal. Its more of an algorithmic estimation of the actual distributions. Root Dirrelation Divergence doesn't require intermediate steps - its a closed form squaring and mean calculation. This means it will not pick up divergences from normal in the same way KLD does, but it is much faster (~9x) and smoothly identifies differnces in either mean or variance. You're less likely to see "bumpiness" as convergence occurs.

```
%%timeit
Empirical.root_dirrelation_divergence(arr_a,arr_b)
```

```
44.8 µs ± 622 ns per loop (mean ± std. dev. of 7 runs, 10,000 loops each)
```

```
%%timeit
Empirical.kl_divergence(arr_a,arr_b,bins=100)
```

```
393 µs ± 8.31 µs per loop (mean ± std. dev. of 7 runs, 1,000 loops each)
```

What Divariance/Root Dirrelation Divergence is not useful for

One subtlety (or maybe glaringly obvious to some) that should be addressed. This measure as I've presented it applies to comparing multiple samples of two classes/types over a single feature:

single feature/multiple samples/two types.

It is not a proxy for KL Divergence when applies over multiple features. Thus when categorizing sales and detetrming if the proportion of categories are out of line with each other, this won't work. It also won't directly work when comparing the semantic embedding of text and the corresponding model output over the entire feature set… that is:

multiple features/one sample/two types

You can compare multiple features/multiple samples/two types using Frechet Inception Distance. It can be related to the above work, but multi-feature divergence isn't addressed here.

Tying together distance and divergence

Divariance is a way to tie together Frechet distance and KL Divergence, two key tools for data scientists for differentiating distributions of data: whether model results and reality or features of classes. Root Dirrelation Divergence an act as a proxy for KL Divergence when comparing samples of normal-like data. It closely mirrors the numerical value and offers advantages of speed, simplicity and convex smoothness. I used to for sufficient feature selection of CLIP embeddings for image/text classification, but I believe it can be useful in others instance where you need a metric for identifying divergence in data sets, specifically data drift, time series (ie stock returns) and distributional projections (weather). Reach out with any questions here in the comments here or through my contact information on github.

Citations:

Proof: partition of variance into expected values: https://statproofbook.github.io/P/var-mean
Proof: partition of co-variance into expected values: https://statproofbook.github.io/P/cov-mean
Proof: KL distribution of two normal distributions: https://statproofbook.github.io/P/norm-kl.html

KL divergence and distance
https://www.johndcook.com/blog/2017/11/08/why-is-kullback-leibler-divergence-not-a-distance/

https://www.stat.uchicago.edu/~lekheng/work/probdist.pdf

Information Gain
https://www.geeksforgeeks.org/machine-learning/information-gain-and-mutual-information-for-machine-learning/

Frechet Distance

Frechet Applied to Multi-variate normal distributions. Dowson & Landau. 1982. JOURNAL OF MULTIVARIATE ANALYSIS 12, 450-455 (1982)
https://www.sciencedirect.com/science/article/pii/0047259X8290077X

1. FID was introduced by Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler and Sepp Hochreiter in
"GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium"

Sufficient Dimension Reduction:

https://jmlr.org/papers/volume3/globerson03a/globerson03a.pdf

Sufficient dimension reduction based on the Hellinger
integral: a general, unifying approach
Xiangrong Yin∗ Frank Critchley† Qin Wang‡
June 7, 2010
https://university.open.ac.uk/stem/mathematics-and-statistics/sites/www.open.ac.uk.stem.mathematics-and-statistics/files/files/Hellinger-submitted.pdf