The effect of anisotropic node splits on Wavelet decomposition of RF

Asaf Abas Uria Mor

02/09/2018

Single Decision Tree

Settings:
Given a sample set of a real (or vector) valued function f on some convex domain $\Omega_0 \in \mathbb{R}^n$: $\{x_i \in \Omega_0, f(x_i)\}_{i \in I}$.
Our goal is to find efficient representation of the underlying function f, overcoming the complexity, geometry and (usually) non-smoothness nature of the function in the sampled points.

Single Decision Tree

We divide Ω_0 into two subdomains, by intersecting it with a hyperplane. The subdivision is performed to minimize a given cost function. We recursively repeat the subdivision process on each of the subdomains until some stopping criteria is met and which defines the leafs of the tree.

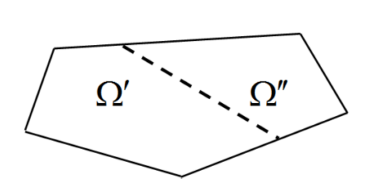

Figure: Subdivision illustration

Cost function

One possible choice of cost function is one such minimizes the variance in each subdomain.
Given a convex domain Ω (associated with a node in the tree) find:

(*i*) A partition by a hyper-plane into two convex subdomains Ω', Ω'', so $\Omega = \Omega' + \Omega''$

(*ii*) Two multivariate polynomials $Q_{\Omega'}, Q_{\Omega''}$ of fixed total degree $r - 1$ (typically low, usually $r = 1$)

The subdomains Ω', Ω'' and the polynomials $Q_{\Omega'}, Q_{\Omega''}$ are chosen to minimize

$$\sum_{x_i \in \Omega'} \left|f(x_i) - Q_{\Omega'}(x_i)\right|^p + \sum_{x_i \in \Omega''} \left|f(x_i) - Q_{\Omega''}(x_i)\right|^p$$

Where $1 < p \leq \infty$ (usually $p = 2$) also, in our case, Q_{Ω} will denote the mean value of Ω points.

Minimizing the cost function

The problem can be formulated as an optimization problem:
find $\theta_0, \ldots, \theta_d$ which minimize

$$\sum_{x_i \in \Omega'} \left| f(x_i) - \overline{f(\Omega')} \right|^2 + \sum_{x_i \in \Omega''} \left| f(x_i) - \overline{f(\Omega'')} \right|^2$$

where $\Omega' = \left\{ x = (x_1, \ldots, x_d) \in \Omega; \sum_{i=1}^{d} \theta_i x_i < \theta_0 \right\}$ and $\Omega'' = \Omega \setminus \Omega'$
Note that the search space of this problem is far from ideal - slightly changing one of the θs can (and will) result a massive jump in the cost function values. Moreover, since the cost function is not continuous, we can't use standard optimization algorithms (CG, Newton, GD etc...)

The division challenge

The number of possible hyperplanes which will divide a region containing n data samples over d features is $\binom{n}{d}$ at worst. Furthermore, we can't simply

In many applications of decision trees, searching through all possible subdivisions is impractical. Thus, in practice - most prominent implementations of DT (scikit-learn&R included) restrict the search to subdivisions by an isotropic hyperplane, that is, splits parallel to one of the features.

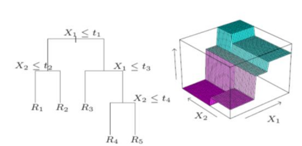

Figure: Isotropic split

Our work here aims to find a near optimal anisotropic split yielding better progress (and thus better approximates the data) compared to isotropic splits

Our approach

Since it is not feasible to go through all possible subdivisions, we turn to heuristics that will (hopefully) find a near optimal hyper-plane which (fingers crossed) outperforms the isotropic split. We propose three approaches built one on top of the other:

1. **Capture the geometry** as if there is **no problem**
2. **Solve the problem** as if there is **no geometry**
3. **Capture the geometry of the problem**

Our approach

Since it is not feasible to go through all possible subdivisions, we turn to heuristics that will (hopefully) find a near optimal hyper-plane which (fingers crossed) outperforms the isotropic split. We propose three approaches built one on top of the other:

1. **Capture the geometry** as if there is **no problem**
    1. Run KMeans ($k = 2$) on the outcomes - Y and label each sample with the predicted label
    2. Use SVM to best separate the labeled data in the features space
2. **Solve the problem** as if there is **no geometry**
3. **Capture the geometry of the problem**

Our approach

Since it is not feasible to go through all possible subdivisions, we turn to heuristics that will (hopefully) find a near optimal hyper-plane which (fingers crossed) outperforms the isotropic split. We propose three approaches built one on top of the other:

1. **Capture the geometry** as if there is **no problem**
   1. Run KMeans ($k = 2$) on the outcomes - Y and label each sample with the predicted label
   2. Use SVM to best separate the labeled data in the features space
2. **Solve the problem** as if there is **no geometry**
   1. Reduce the dimensionality of X space (PCA)
   2. Use first two PCs as starting points for standard gradient-free optimization (Nedler-Mead solver)
3. **Capture the geometry of the problem**

Our approach

Since it is not feasible to go through all possible subdivisions, we turn to heuristics that will (hopefully) find a near optimal hyper-plane which (fingers crossed) outperforms the isotropic split. We propose three approaches built one on top of the other:

1. **Capture the geometry** as if there is **no problem**
   1. Run KMeans ($k = 2$) on the outcomes - Y and label each sample with the predicted label
   2. Use SVM to best separate the labeled data in the features space
2. **Solve the problem** as if there is **no geometry**
   1. Reduce the dimensionality of X space (PCA)
   2. Use first two PCs as starting points for standard gradient-free optimization (Nedler-Mead solver)
3. **Capture the geometry of the problem**
   1. Try to find p - the multidimensional direction in the X space that explains the maximum multidimensional variance direction in the Y space (PLS)
   2. Find H - the hyperplane which p is it's normal
   3. Find best offset of H

Toy demo

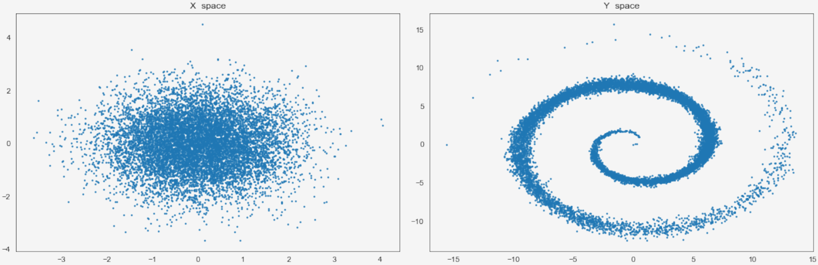

Figure: Iso splits

Toy demo

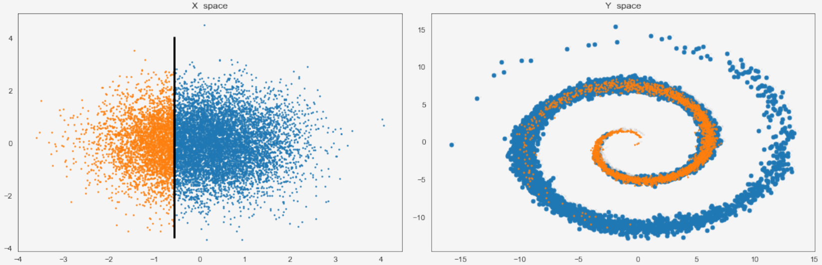

Figure: Iso splits

Toy demo

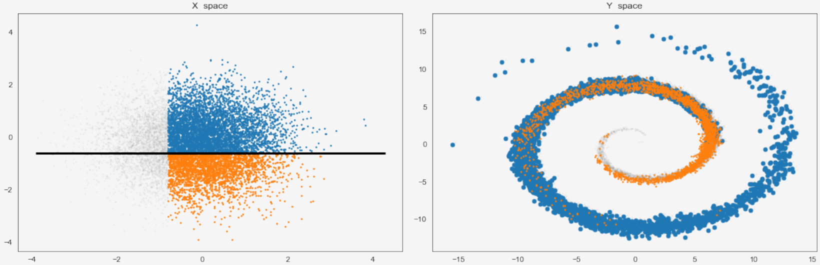

Figure: Iso splits

Toy demo

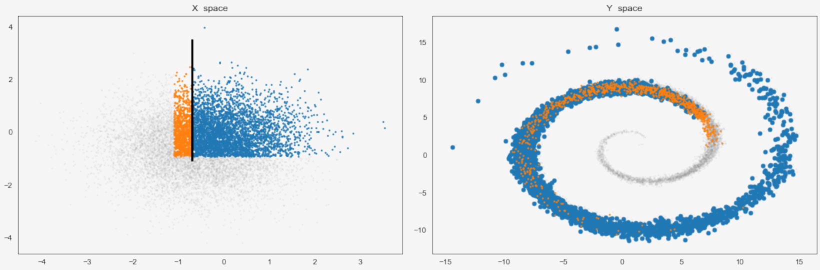

Figure: Iso splits

Toy demo

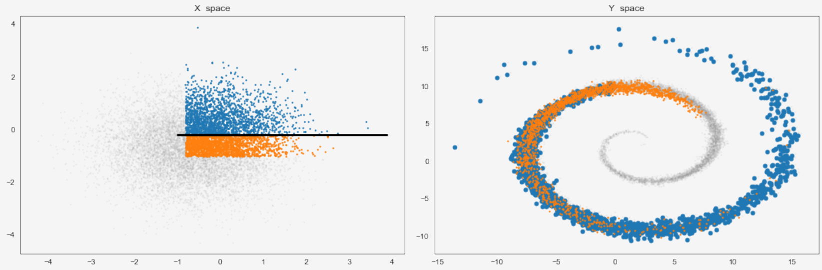

Figure: Iso splits

Toy demo

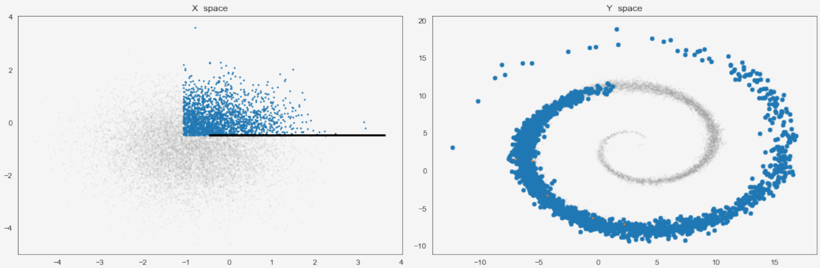

Figure: Iso splits

Toy demo

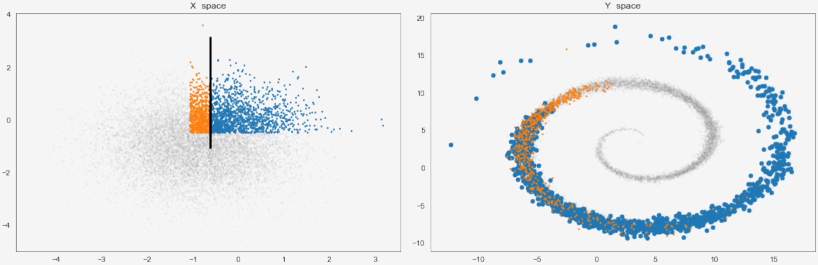

Figure: Iso splits

Toy demo

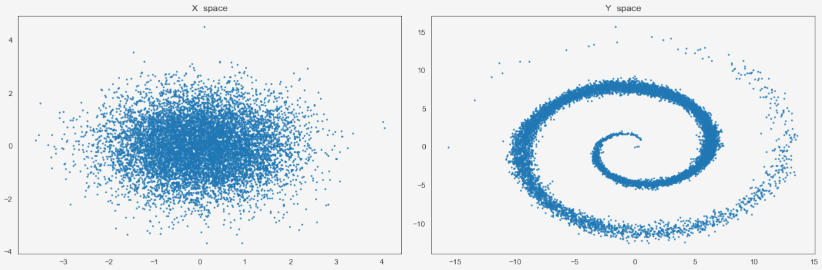

Figure: 2Means

Toy demo

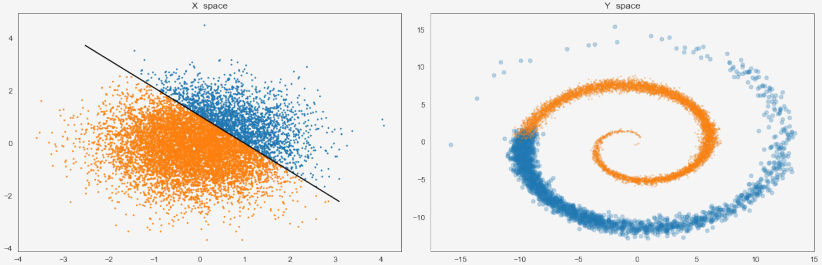

Figure: 2Means split

Toy demo

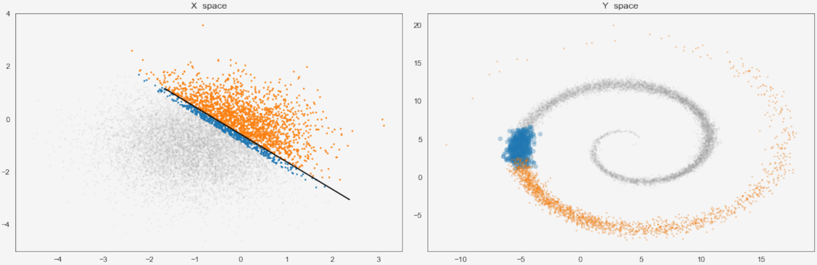

Figure: 2Means split

Toy demo

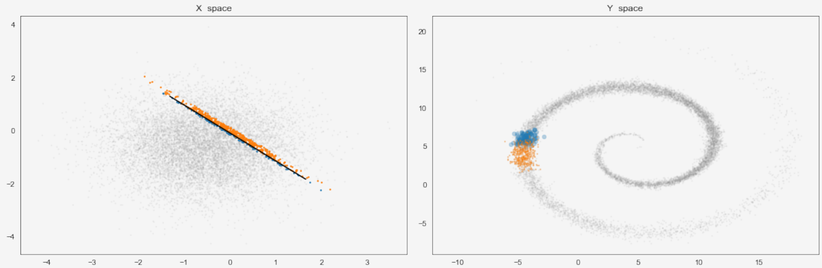

Figure: 2Means split

Toy demo

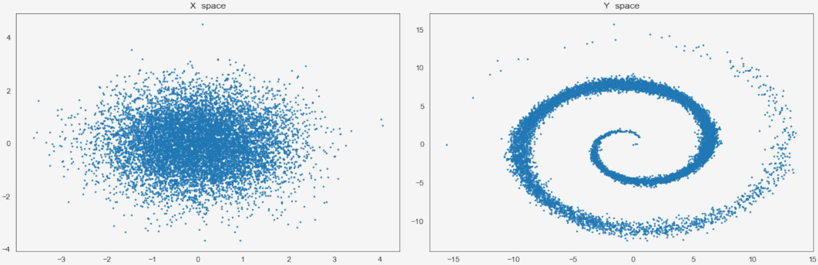

Figure: PCA

Toy demo

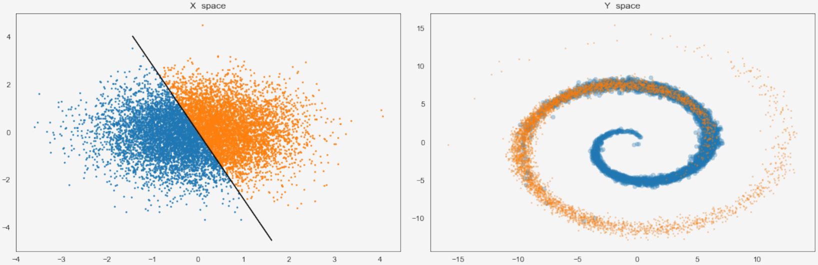

Figure: PCA

Toy demo

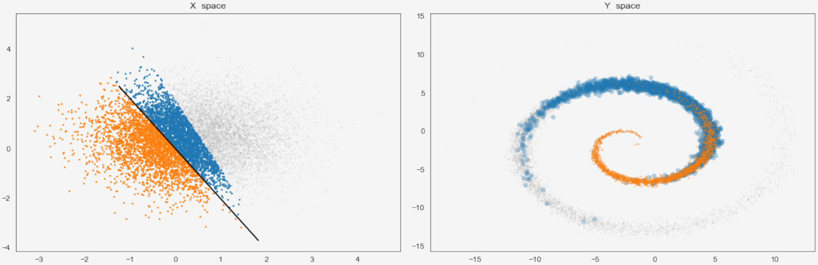

Figure: PCA

Toy demo

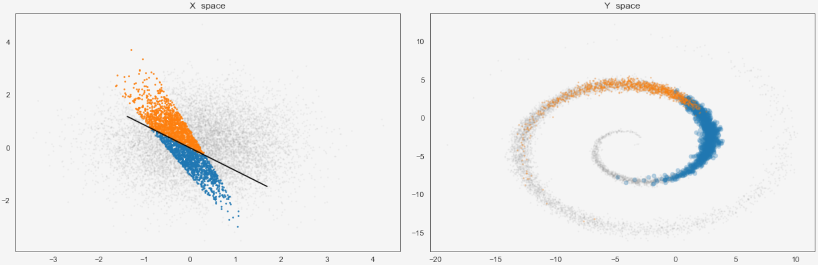

Figure: PCA

Toy demo

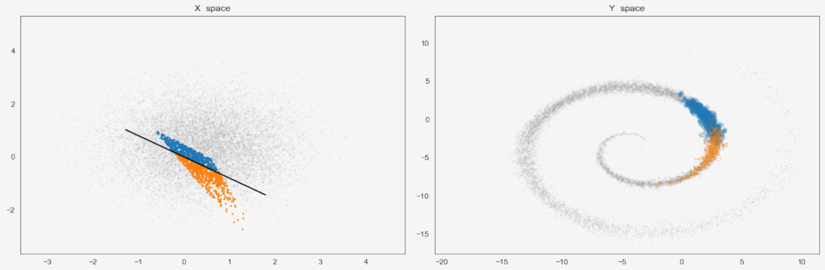

Figure: PCA

Toy demo

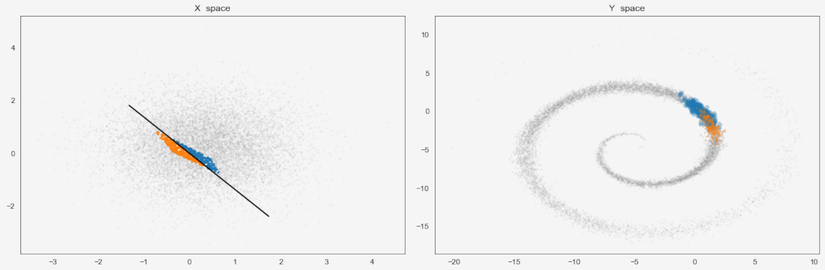

Figure: PCA

Toy demo

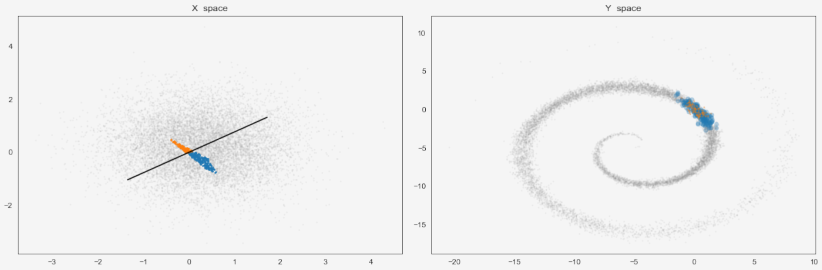

Figure: PCA

Toy demo

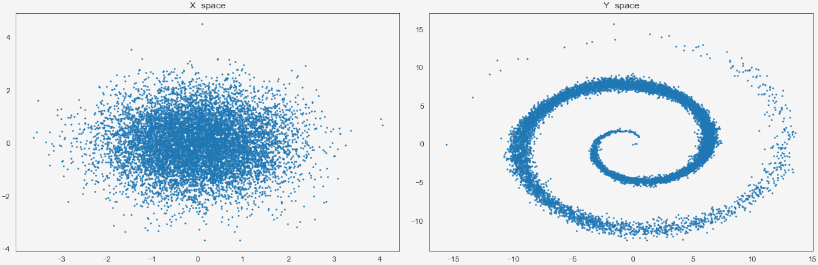

Figure: pls **NotImplementedError**

Toy demo

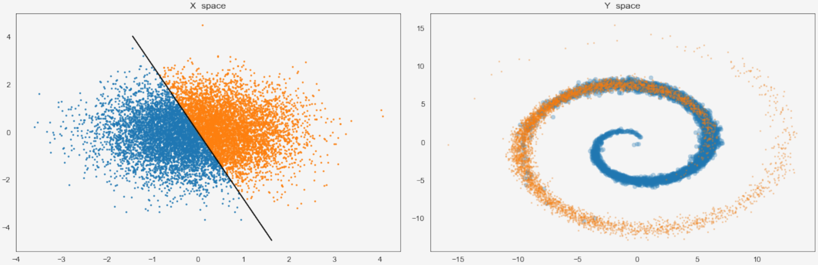

Figure: pls **NotImplementedError**

Toy demo

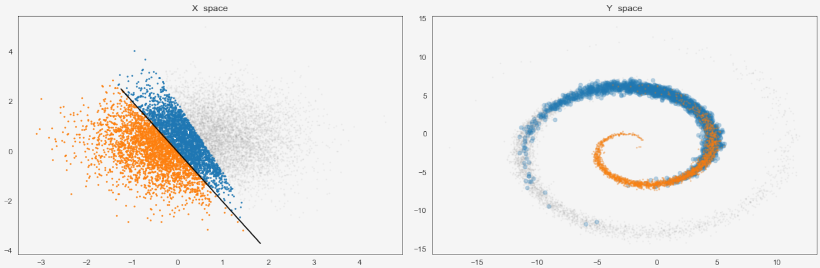

Figure: pls **NotImplementedError**

Toy demo

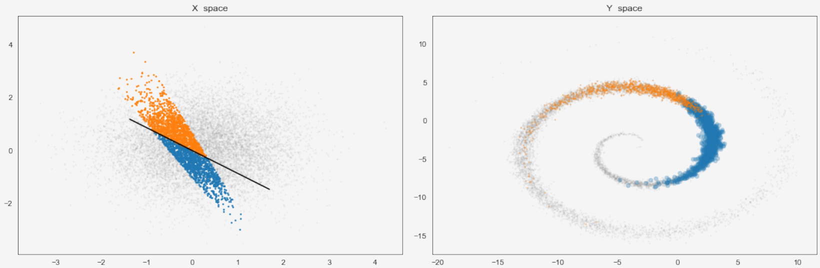

Figure: pls **NotImplementedError**

Toy demo

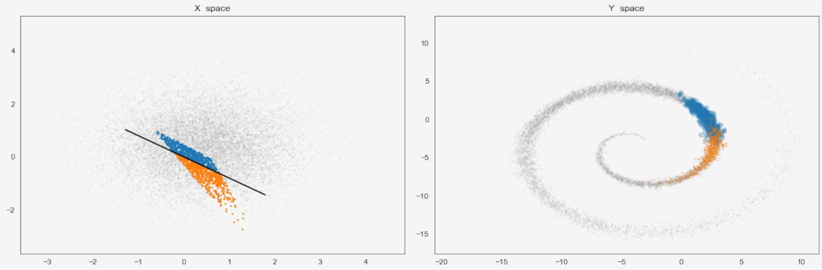

Figure: pls **NotImplementedError**