On the Use of Random Forest for Two-Sample Testing, with Applications in Empirical Finance

Simon Hediger and Jeffrey Näf

University of Zurich and ETH

June 25, 2019

Joint Work with Loris Michel

Outline

Outline

Two-Sample Test

We assume to encounter a collection of random vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$ and $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ with support $\mathcal{X} \subset \mathbb{R}^p$ and $\mathcal{Y} \subset \mathbb{R}^p$ respectively, such that $\mathbf{X}_i \overset{iid}{\sim} \mathbb{P}_X$ and $\mathbf{Y}_i \overset{iid}{\sim} \mathbb{P}_Y$, where $\mathbb{P}_X$ and $\mathbb{P}_Y$ are some probability measures on $\mathbb{R}^p$.
Given this sample, we want to test

$$H_0 : \mathbb{P}_X = \mathbb{P}_Y, \quad H_A : \mathbb{P}_X \neq \mathbb{P}_Y.$$

Binomial Test

Given these iid samples of n vectors, we give each $\mathbf{X}_i$ the label of one, while each $\mathbf{Y}_i$ is labeled zero. Based on this dataset

$$D_{2n} = \begin{pmatrix} 1 & \mathbf{X}_1 \\ & \vdots \\ 1 & \mathbf{X}_n \\ 0 & \mathbf{Y}_1 \\ & \vdots \\ 0 & \mathbf{Y}_n, \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{Z}_1 \\ & \vdots \\ 1 & \mathbf{Z}_n \\ 0 & \mathbf{Z}_{n+1} \\ & \vdots \\ 0 & \mathbf{Z}_{2n}, \end{pmatrix},$$

we fit a binary random forest classifier, giving a discrimination function, $g(\cdot, \boldsymbol{\theta}, D_{2n}) : \mathbb{R}^p \to \{0, 1\}$.

Binomial Test

For a training set of size $2n$ and a test set of size m_n, we then calculate

$$\hat{L}_{m_n} = \frac{1}{m_n} \sum_{i=1}^{m_n} \underbrace{\mathbb{I}_{\{g(\mathbf{Z}_i, D_{2n}, \boldsymbol{\theta}) \neq \ell_i\}}}_{\varepsilon_i}.$$

$\hat{L}_{2n}$ serves as an estimate for L_{2n}, the true unknown generalization error for a given dataset:

$$L_{2n} := P(g(\mathbf{Z}, D_{2n}, \boldsymbol{\theta}) \neq \ell | D_{2n}).$$

Binomial Test

Since under H_0

$$\varepsilon_i \overset{\text{iid}}{\sim} \text{Bern}(1/2)$$

and hence

$$m_n \hat{L}_{m_n} = \sum_{i=1}^{m_n} \varepsilon_i \sim \text{Binomial}(m_n, 1/2),$$

we are able to construct an exact test, by simply rejecting if

$$m_n \hat{L}_{m_n} < B^{-1}(\alpha),$$

where $B^{-1}(\alpha)$ is the α quantile of the Binomial$(m_n, 1/2)$ distribution.*

*This idea was already explored, for different classifiers, in Ramdas et al. (2016) or Lopez-Paz and Oquab (2017)

Permutation Test

Instead of using the out-of-sample-error above, we can also use the OOB-error

$$U_{2n} = h_{2n}((\ell_1, \mathbf{Z}_1), \ldots, (\ell_n, \mathbf{Z}_{2n})) = \frac{1}{2n} \sum_{i=1}^{2n} \mathbb{I}_{\{g(\mathbf{Z}_i, D_{2n} \setminus i) \neq \ell_i\}},$$

where $g(\cdot, D_{2n} \setminus i) : \mathbb{R}^d \to \{0, 1\}$, represents the forest not containing the i^{th} observation for training.

HOWEVER, the behavior of the OOB error is hard to control theoretically, even under H_0, making it unsuited for our purposes at first inspection.

Permutation Test

After calculation of the OOB-error reshuffle the labels K times to obtain K permutations, $\sigma_1, \ldots, \sigma_K$ say.

For each of these new datasets $\left(\mathbf{Z}_i, \ell_{\sigma_j(i)}\right)_{i=1}^{2n}$, calculate the OOB error

$$U_{2n}^{(j)} := h_{2n}((\mathbf{Z}_1, \ell_{\sigma_j(1)}), \ldots, (\mathbf{Z}_{2n}, \ell_{\sigma_j(2n)})).$$

Under H_0, $(\ell_1, \ldots, \ell_{2n})$ and $(\mathbf{Z}_1, \ldots, \mathbf{Z}_{2n})$ are independent and each $U_{2n}^{(j)}$ is simply an iid draw from the distribution F of the random variable $U_{2n}^{(j)}|(\mathbf{Z}_1, \ldots, \mathbf{Z}_{2n})$.

Permutation Test

Finally, approximate the α quantile $F^{-1}(\alpha)$ by performing a large number of permutations, and reject if

$$U_{2n} < F^{-1}(\alpha)$$

.

NOTE: under H_0

$$P(\text{Rejecting } H_0) = \mathbb{E}[P(U_{2n} < F^{-1}(\alpha)|\mathbf{Z}_1, \ldots, \mathbf{Z}_{2n})] \leq \mathbb{E}[\alpha] = \alpha.$$

Permutation Test (Pseudo-Code)

Algorithm 1 hypoRF $\leftarrow$ function$(X, Y, K, ...)$

Require: $X, Y \in \mathbb{R}^{n \times p}$ and $K \in \mathbb{N}$ // $p > n$ is not an issue

$\ell \leftarrow (1, ..., 1, 0, ..., 0)'$ // ℓ represents the response variable

$Z \leftarrow [X \quad Y]_{2n \times p}$ // row bind X and Y

$D_{2n} \leftarrow (\ell_i, \mathbf{Z}_i)_{i=1}^{2n}$

$g(., D_{2n}) \leftarrow rf$ // training of a Random Forest classifier

$OOB \leftarrow \frac{1}{2n} \sum_{i=1}^{2n} \mathbb{I}_{\{g(\mathbf{Z}_i, D_{2n}^{-i}) \neq \ell_i\}}$ // calculating the OOB-error

for `j in 1:K` **do**

  $D_{2n}^{j} \leftarrow \left(\ell_{\sigma_j(i)}, \mathbf{Z}_i\right)_{i=1}^{2n}$ // reshuffle the label

  $OOB^j \leftarrow \frac{1}{2n} \sum_{i=1}^{2n} \mathbb{I}_{\{g(\mathbf{Z}_i, D_{2n}^{j,-i}) \neq \ell_{\sigma_j(i)}\}}$ // calculating the OOB-error

end for

$\mu \leftarrow \frac{1}{K} \sum_{j=1}^{K} OOB^j$

$\sigma^2 \leftarrow \frac{1}{K-1} \sum_{j=1}^{K} (OOB^j - \mu)^2$

$pvalue \leftarrow \mathbb{P}(OOB < \Phi^{-1}(\alpha) \cdot \sigma + \mu)$ // using a normal approximation and $\alpha = 0.05$

return *pvalue*

Outline

Powerful Two-Sample Tests

As a comparison we will use 3 kernel-based tests:

1. the "quadratic time MMD" (Gretton et al., 2012a) using a permutation approach to approximate the H_0 distribution ("MMDboot")
2. its optimized version "MMD-full" †
3. as well as the "ME" test with optimized locations, "ME-full" (Jitkrittum et al., 2016)

†The original idea for this was formulated in Gretton et al. (2012b), however they subsequently used a linear version of the MMD. We instead use the approach of Jitkrittum et al. (2016), which uses the optimization procedure of Gretton et al. (2012b) together with the quadratic MMD from Gretton et al. (2012a).

Gaussian Mean Shift in Some Columns

Consider

$$\mathbb{P}_X = N(\boldsymbol{\mu}_1, I_{p\times p}) \text{ and } \mathbb{P}_Y = N(\boldsymbol{\mu}_2, I_{p\times p})$$

so that the testing problem reduces to

$$H_0 : \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 \quad \text{vs} \quad H_1 : \boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2.$$

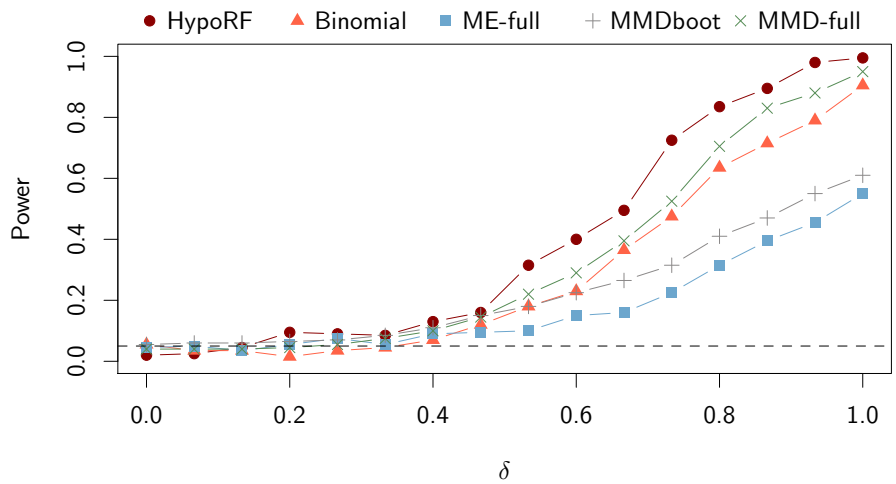
We induce a shift of size $\delta \in \mathbb{R}$ in $d < p$ elements of $\boldsymbol{\mu}_2$

$$\boldsymbol{\mu}_2 = \boldsymbol{\mu}_1 + (\delta/\sqrt{d}) \cdot \mathbf{1}_d{}^{\ddagger}$$

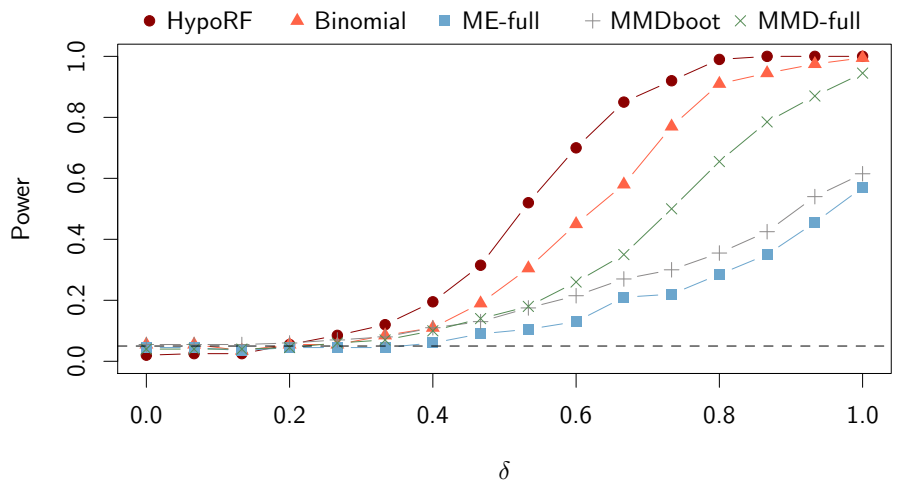
We study a "moderately sparse" case $d = 20$ (10% out of $p = 200$) and a "sparse" case $d = 2$ (1% out of $p = 200$).

$^{\ddagger}\mathbf{1}_d$ is a p dimensional vector with the first d elements equal to 1 and the remaining $p - d$ elements equal to 0

Gaussian Mean Shift in Some Columns



(a) $d = 20$, moderately sparse case.



(b) $d = 2$ sparse case.

Figure: A point in the figures represents a simulation of size $S = 200$ for a specific test and a $\delta \in (0, 0.125, 0.25, ..., 1)$. Each of the $S = 200$ simulation runs we sampled $n = 300$ observations from a $p = 200$ dimensional multivariate Gaussian distribution, where d columns have a shift in mean of $\frac{\delta}{\sqrt{d}}$ and likewise $n = 300$ observations from $p = 200$ independent standard normal distributions. The Random Forest used 600 trees and a minimal node size to consider a random split of 4.

Contamination Example

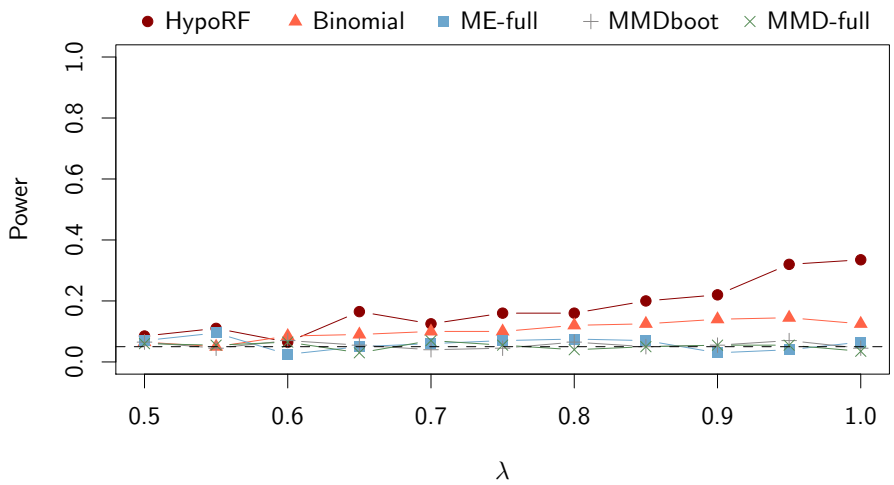
Let $\mathbb{P}_X = N(\boldsymbol{\mu}, \Sigma)$ with $\boldsymbol{\mu}$ set to $50 \cdot \mathbf{1}$ and $\Sigma = 25 \cdot I_{p \times p}$. For the alternative, we consider the mixture

$$\mathbb{P}_Y = \lambda \mathbb{P}_c + (1 - \lambda)\mathbb{P}_X,$$

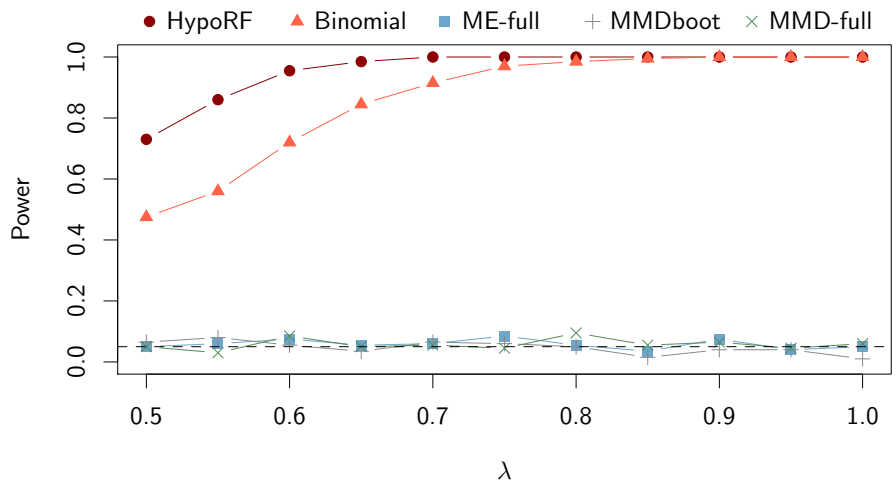
$\lambda \in [0, 1]$, and $\mathbb{P}_c$ some distribution on $\mathbb{R}^p$. This is what we describe as a "contamination" of $\mathbb{P}_X$ by $\mathbb{P}_Y$ with λ determining the contamination strength.

We take $\mathbb{P}_c$ to be another independent $(p - d)$-variate Gaussian together with d components that are in turn independent Binomial$(100, 0.5)$ distributed.

Contamination Example



(a) $d = 20$, sparse case.



(b) $d = p$, no sparsity.

Figure: A point in the figure represents a simulation of size $S = 200$ for a specific test and a $\lambda \in (0.5, 0.55, ..., 1)$. Each of the $S = 200$ simulation runs we sampled $n = 300$ observations from the contaminated distribution with $\lambda \in (0.5, 0.55, ..., 1)$ and likewise $n = 300$ observations from $p = 200$ independent standard normal distributions. The Random Forest used 600 trees and a minimal node size to consider a random split of 4.

Outline

Changing the Dependency Structure

Consider

$$\mathbb{P}_X = N(0, I_{p\times p}) \text{ and } \mathbb{P}_Y = N(0, \Sigma),$$

where Σ is some positive definite correlation matrix.

For simplicity, we only consider a single correlation number ρ, which we use in all $p(p-1)/2$ unique correlation coefficients in Σ.

Changing the Dependency Structure

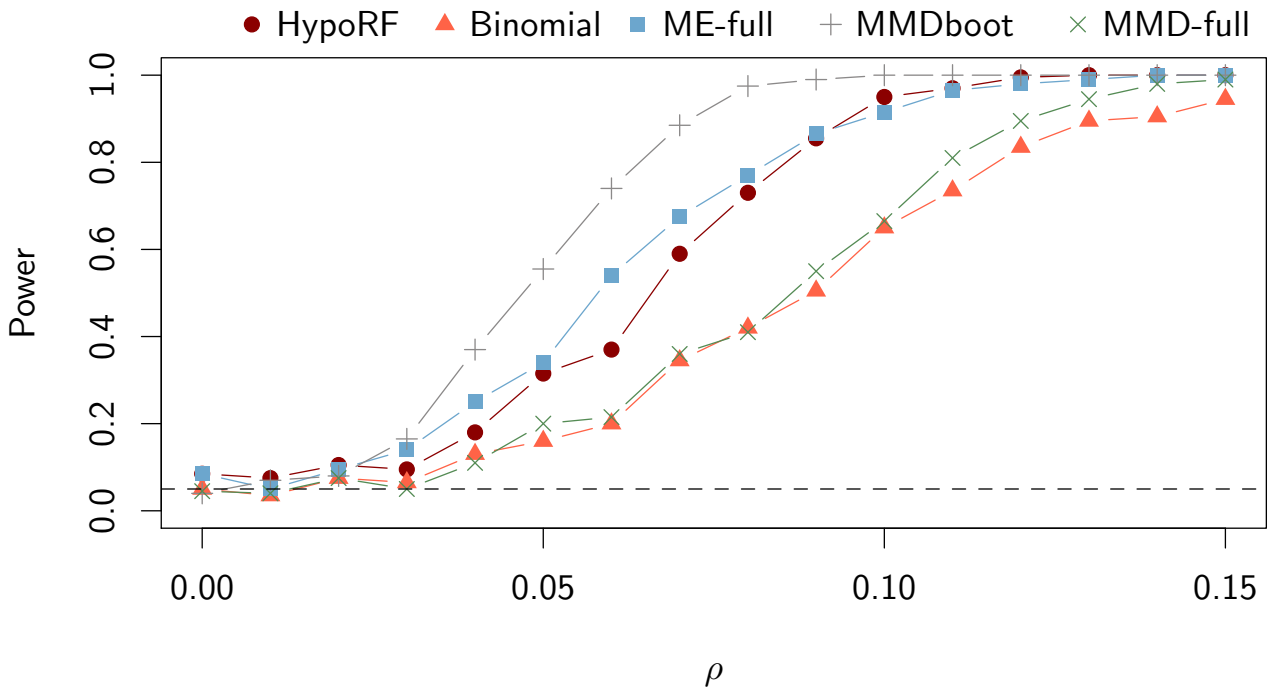


Figure: A point in the figure represents a simulation of size $S = 200$ for a specific test and a $\rho \in (0, 0.01, 0.02..., 0.15)$. Each of the $S = 200$ simulation runs we sampled $n = 300$ observations from a $p = 60$ dimensional multivariate normal distribution with $\rho \in (0, 0.01, 0.02..., 0.15)$, representing $\mathbb{P}_Y$. Likewise $n = 300$ observations were sampled from a $p = 60$ dimensional multivariate normal distribution using $\rho = 0$, representing $\mathbb{P}_X$. The Random Forest used 600 trees and a minimal node size to consider a random split of 4.

Outline

Multivariate Normal Mean-Variance Mixture Distribution

The random vector $\mathbf{Y}$ is said to have a multivariate normal mean-variance mixture distribution (MNMVM) if

$$\mathbf{Y} = \mathbf{m}(G) + \mathbf{H}(G)^{1/2}\mathbf{Z},$$

where

- $\mathbf{Z} \sim \mathrm{N}(\mathbf{0}, \mathbf{I}_K)$;
- $G \geq 0$ is a non-negative, univariate random variable which is independent of $\mathbf{Z}$;
- $\mathbf{H} : [0, \infty) \to \mathbb{R}^{K \times K}$ is a measurable function that returns a symmetric, positive definite $K \times K$ matrix;
- $\mathbf{m} : [0, \infty) \to \mathbb{R}^K$ is a measurable function.

The name MNMVM comes from the fact that:

$$\mathbf{Y} \mid (G = g) \sim \mathrm{N}(\mathbf{m}(g), g\mathbf{H}).$$

Multivariate Normal Mean-Variance Mixture Distribution

The random vector $\mathbf{Y}$ is said to have a multivariate normal mean-variance heterogeneous tails mixture distribution (MNHMVM) if it can be expressed as

$$\mathbf{Y} = \mathbf{m}(\mathbf{G}) + \mathbf{H}(\mathbf{G})^{1/2}\mathbf{Z},$$

- $\mathbf{Z} \sim \mathrm{N}(\mathbf{0}, \mathbf{I}_K)$;
- $\mathbf{G} = (G_1, \ldots, G_K)$ is a vector of *independent* non-negative, univariate random variable which is independent of $\mathbf{Z}$;
- $\mathbf{H} : [0, \infty)^K \to \mathbb{R}^{K \times K}$ is a measurable function that returns a symmetric, positive definite $K \times K$ matrix;
- $\mathbf{m} : [0, \infty)^K \to \mathbb{R}^K$ is a measurable function.

The name comes from the fact that $\mathbf{Y} \mid (\mathbf{G} = \mathbf{g}) \sim \mathrm{N}(\mathbf{m}(\mathbf{g}), \mathbf{H}(\mathbf{g}))$.

COMFORT and HGH

- An extension of the MNMVM approach was used in Paolella and Polak (2015) to build the powerful COMFORT[§] model.
- The MNHMVM approach was introduced and used in Näf et al. (2018) to build the so-called HGH model.
- Both models can be estimated efficiently by an EM type algorithm and both allow for a very efficient way to conduct PF optimization.
- Important for our purposes: Both models incorporate latent random variables, given by the "G" or "$\mathbf{G}$". In joint asset model, these can be interpreted as unobserved "shocks" to the assets.

[§] A Common Market Factor Non-Gaussian Returns Model

COMFORT and HGH

- Viewed through this lens, the models represent two extremes; on the one side, there is just one latent shock (COMFORT), while on the other there are K (independent) latent shocks (HGH).
- Is it possible to correctly identify whether one or several shocks to a set of assets are more appropriate?
- In other words; does it make sense to give each asset its own shock, or have one shock jointly for several assets?
- This problem is not easy. The number of G's involved changes the dependency structure in a nontrivial way.
- However, simulation shows that our test is able to differentiate between a model having one or K latent variables or something in between.

Application Idea

- We reduce the problem by fixing some parameters in the distribution of the latent shocks, such that each G, whether we have several or just one, has a Gamma$(\lambda, 1)$ distribution.
- This means marginally, we have K variance gamma random variables,

$$Y_k \sim VG(\lambda_k, \mu_k), \text{ for } k = 1, \ldots, K,$$

where

$$f_{VG}(\lambda, \mu) = \frac{2}{\sqrt{2\pi}\Gamma(\lambda)} \left(\frac{|x-\mu|}{\sqrt{2}} \right)^{\lambda - 1/2} K_{\lambda-1/2}(\sqrt{2}|x-\mu|)$$

the density of a variance gamma with modified Bessel function $K_{\lambda-1/2}$ (see Paolella (2007) Section 9.2).

Goal

- Let us first assume the Y_k, $k = 1, \ldots, K$ are independent.
- Instead of getting K different estimates, $\hat{\lambda}_1, \ldots, \hat{\lambda}_K$, we identify $J < K$ groups, such that for a given group all λ's are the same.

We do this for a set of n observations, by optimizing the penalized log-likelihood

$$S(\lambda_1, \ldots, \lambda_K) := \sum_{k=1}^{K} \sum_{i=1}^{n} \log(L_{VG}(\lambda_k, \mu_k, y_i)) + \eta \sum_{j=1}^{K-1} \sum_{j>i} |\lambda_j - \lambda_i|$$

After Throwing the LASSO

- For a given penalty η, the components with equal λ's form a natural grouping.
- These groups fulfill the necessary condition of having equal parameters for the latent shocks.
- However, since equal λ's do not necessarily imply equal G's, this is not enough!

After Throwing the LASSO

- For each group $j = 1, \ldots, J$ of size k_j, efficiently estimate the parameters of the COMFORT model if $k_j > 1$ and do nothing if $k_j = 1$.
- For each j, test the real data against data simulated from the COMFORT model with parameters estimated above.
- If the test cannot reject, we are satisfied and keep the k_j assets as one group.
- For all groups for which the test rejects equality in distribution, we repeat the procedure with a *lower* penalization η.

Algorithm Sketch

1. Set η (at the beginning "very" high to get only one group)
2. Optimize $S(\lambda_1, \ldots, \lambda_K)$ with respect to $\lambda_1, \ldots, \lambda_K$
3. Group all assets that have the same λ_j. This gives $J < K$ groups.
4. For each group $j = 1, ..., J$ use our two-sample test to check whether we cannot differentiate between the real data in group j and simulated data from the k_j-variate COMFORT model with $\hat{\lambda}_j$ as common parameter.
5. If the test cannot reject, we have found a group with a common λ_j.
6. For the groups for which we reject, reduce the penalty η and repeat the above steps.

Stop If: we can not reject the two-sample test for all the groups **OR** if there are only "groups" with $k_j = 1$ left.

High Hopes

- We hope that we are able to identify the right latent structure in simulations.
- It might even be possible to prove consistency under some reasonable assumption, i.e. that we are able to identify the right latent structure with high probability when n increases.
- We also hope to use it successfully on real data, though some more tweeks might be necessary.
- In general the algorithm is constructed in such a way that the model is parsimonious: Only introduce a new latent variable/group if it is absolutely necessary.

Conclusion

- An easy to use two-sample test for high dimensions
- Many application possibilities (not just in finance)
- Future research: Modify the test such that it is applicable beyond two-sample testing (testing multiple groups at once, distribution testing, . . .)

Bibliography

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012a). A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(1):723–773.

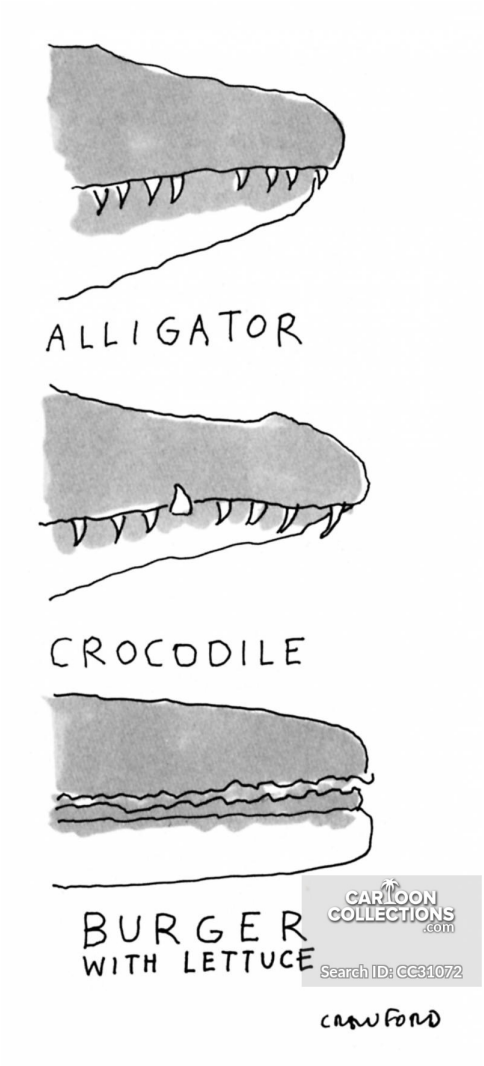
Gretton, A., Sejdinovic, D., Strathmann, H., Balakrishnan, S., Pontil, M., Fukumizu, K., and Sriperumbudur, B. K. (2012b). Optimal Kernel Choice for Large-Scale Two-Sample Tests. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1205–1213. Curran Associates, Inc.

Jitkrittum, W., Szabó, Z., Chwialkowski, K. P., and Gretton, A. (2016). Interpretable Distribution Features with Maximum Testing Power. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 181–189. Curran Associates, Inc.

Lopez-Paz, D. and Oquab, M. (2017). Revisiting Classifier Two-Sample Tests.

Outline

Classification (Supervised Learning)



According to which attributes (variables) do you decide in which class a certain observation belongs?

Classification (Supervised Learning)

The task can be summarized as:
Classifying observations according to optimized rules into a set of categories.

The ingredients to get a classifier are

- an $n \times p$ dimensional data set containing the predictor variables and the label: $D_n = (X, \ell)$
- a decision function with parameters $\boldsymbol{\theta}$: $g(\boldsymbol{x}, \boldsymbol{\theta})$
- a cost function to optimize with respect to $\boldsymbol{\theta}$: $h_n(g(\boldsymbol{x}, \boldsymbol{\theta}), D_n)$

Classification Tree: Two-Dimensional Toy Example

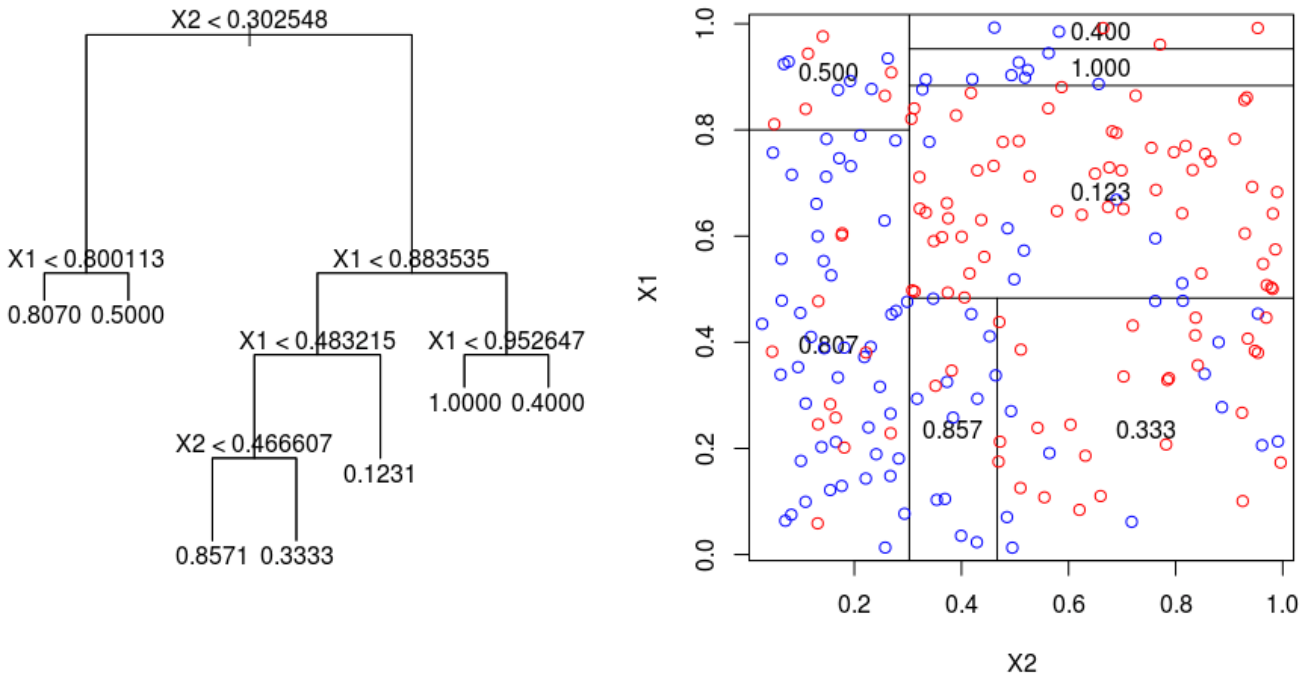


Figure: A simple illustration of a classification tree

Source: `https://www.datacamp.com/community/tutorials/decision-trees-R`

Classification Tree

The underlying model function for a classification tree is

$$g(\boldsymbol{x}, \boldsymbol{\beta}) = \sum_{r=1}^{M} \beta_r \mathbb{I}_{\{\boldsymbol{x} \in \mathcal{R}_r\}},$$

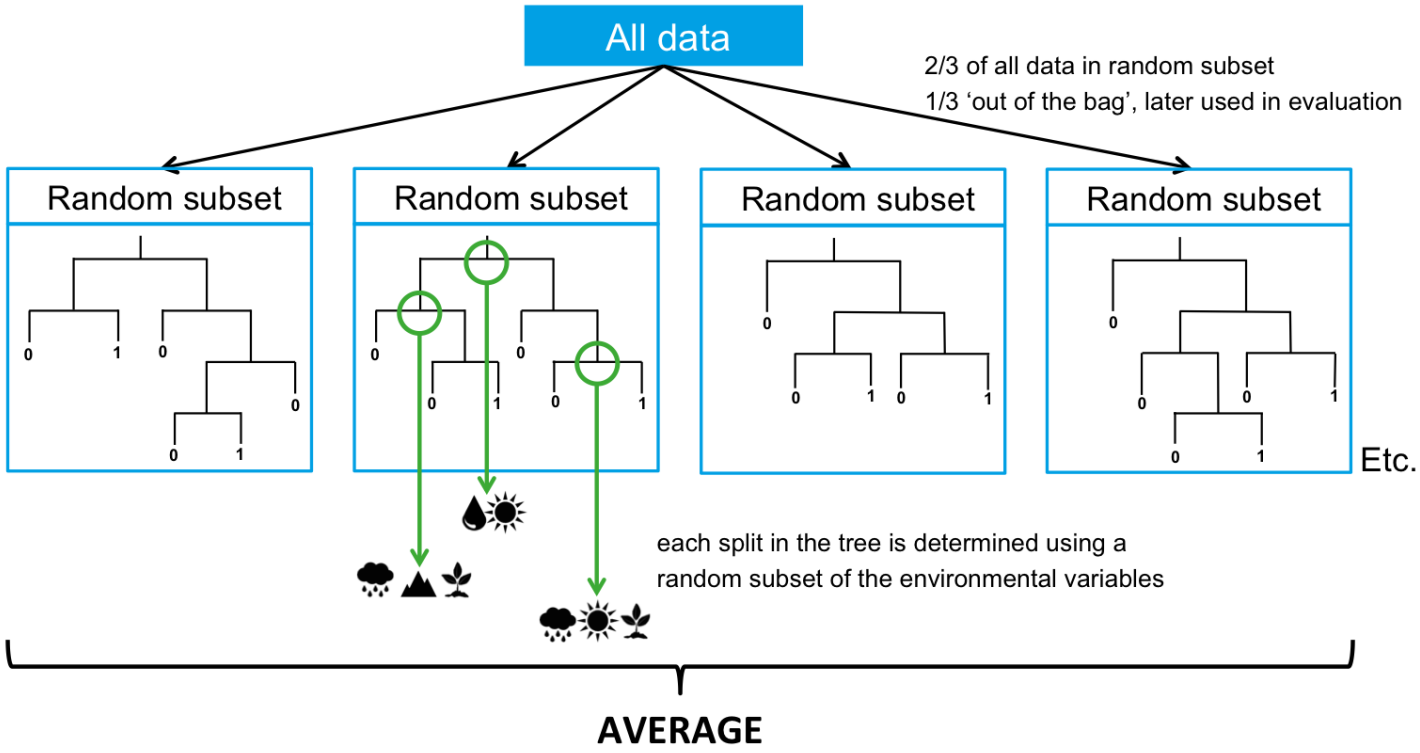
where $\mathcal{P} = \{\mathcal{R}_1, \ldots, \mathcal{R}_M\}$ is a partition of $\mathbb{R}^p$.

For a given partition of $\mathbb{R}^p$ we estimate β_r by

$$\hat{\beta}_r = \frac{\sum_{i=1}^{n} \ell_i \mathbb{I}_{\{\boldsymbol{x}_i \in \mathcal{R}_r\}}}{\sum_{i=1}^{n} \mathbb{I}_{\{\boldsymbol{x}_i \in \mathcal{R}_r\}}}$$

How do we get a reasonable partition of $\mathbb{R}^p$?
Answer: proceed in a greedy way to find the best splitting variables and best split points.

Random Forest in Classification: Illustration



> find the set of predictor variables that produce the strongest classification model

Figure: Illustration of A Random Forest in Classification

Source: `https://support.bccvl.org.au/support/solutions/articles/6000083217-random-forest`

Random Forest in Classification

We assume a collection of classifiers $\{g(\mathbf{x}, \theta_k)\}_{k=1}^K$ as in Breiman (2001), where θ_k are iid realizations of a random element θ.

For a given realization of $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$ (iid copies of θ), this leads to a classifier $g(\mathbf{x}, \boldsymbol{\theta}, D_n)$, where

$$g(\mathbf{x}, \boldsymbol{\theta}, D_n) = \begin{cases} 1, & \text{if } \frac{1}{k}\sum_{j=1}^k g(\mathbf{x}, \theta_j, D_n) \geq 1/2 \\ 0, & \text{if } \frac{1}{k}\sum_{j=1}^k g(\mathbf{x}, \theta_j, D_n) < 1/2 \end{cases}.$$

Since the random forest splits $\mathbb{R}^p$, we then have $g(\boldsymbol{x}, \boldsymbol{\theta}, D_n) : \mathbb{R}^p \to \{0, 1\}$,

i.e. for every $\mathbf{x} \in \mathbb{R}^p$ we can assign a zero or one.

What is an OOB-error?

In general:
When a learning algorithm is trained with a bootstrap sample, the observations not contained in the bootstrap (out-of-bag) are used as the test set.

In Random Forest:
The Random Forest is a collection of trees, each trained with a bootstrap sample. OOB-error is the mean prediction error on each training sample $\boldsymbol{X}_i$, using only the trees that did not have $\boldsymbol{X}_i$ in their bootstrap sample (approximately 1/3 of the trees).
Formally, as the amount of trees goes to infinity

$$h_n((\ell_1, \mathbf{X}_1), \ldots, (\ell_n, \mathbf{X}_n)) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\{g(\mathbf{X}_i, D_n \setminus i) \neq \ell_i\}}$$

Note: We see the OOB-error as a function of the Random Forest.

Appendix: Using the U-Statistic Theory

Algorithm 2 RFTest $\leftarrow$ function$(X, Y, \alpha, ...)$

Require: $X, Y \in \mathbb{R}^{n \times p}$

$\ell \leftarrow (1, ..., 1, 0, ..., 0)' \in \mathbb{R}^{2n}$

$Z \leftarrow [X \quad Y]_{2n \times p}$

$D_{2n} \leftarrow (\ell_i, \boldsymbol{Z_i})_{i=1}^{2n}$

for `k in 1:K` **do**

  $\{S_{k,1}, S_{k,2}\} \leftarrow$ random partition of D_{2n}

  $\bar{h}_{n,k} \leftarrow \frac{1}{2} \sum_{j=1}^{2} h_n(S_{k,j})$

end for

$\hat{U}_{2n,K} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \bar{h}_{n,k}$

$\sigma^2_{WP} \leftarrow \frac{1}{Km(m-1)} \sum_{k=1}^{K} \sum_{j=1}^{m} \left(h_n(S_{k,j}) - \bar{h}_{n,k}\right)^2$

$\sigma^2_{BP} \leftarrow \frac{1}{K} \sum_{k=1}^{K} (\bar{h}_{n,k} - \hat{U}_{2n,K})^2$

$\hat{V}_{2n,K} \leftarrow \sigma^2_{WP} - \sigma^2_{BP}$

if $\hat{V}_{2n,K} < 0$ **then**

  $\hat{V} \leftarrow \sigma^2_{WP}$

else

  $\hat{V} \leftarrow \hat{V}_{2n,K}$

end if

if $\frac{\hat{U}_{2n,K} - 0.5}{\sqrt{\hat{V}}} < \Phi^{-1}(\alpha)$ **then**

  return (reject H0)

else

  return (do not reject H0)

end if

Appendix: Using the U-Statistic Theory

Using the theory derived in Mentch and Hooker (2016) we get that

$$\frac{\hat{U}_{2n,K} - 0.5}{\sqrt{\hat{V}}} \xrightarrow{D} N(0,1),$$

where the variance estimate is derived as in Wang and Lindsay (2014).

Note: We need $K > 2n$ and a reasonable amount of trees.

Appendix: Copula

According to the theorem by Sklar (1959), any multivariate distribution is directly linked to a copula.

Let

$$\boldsymbol{X} = (X_1, ..., X_d)' \sim F$$

with marginal distribution functions $F_1, ..., F_d$ then Sklar (1959) shows that

$$\begin{aligned} F(x_1, ..., x_2) =& C(F_1(x_1), ..., F_d(x_d)), \\ & (x_1, ..., x_d)' \in (\mathbb{R} \cup \{-\infty, \infty\})^d, \end{aligned}$$

where C is a d-dimensional copula.

Appendix: Copula

Consider two samples, X and Y, having cumulative distribution functions F_X and F_Y, respectively.
The test of interest is of the form

$$H0: \ F_X = F_Y \quad \text{vs.} \quad H1: \ F_X \neq F_Y.$$

F_X is a p dimensional multivariate Gaussian, hence

$$X \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_{p \times p}).$$

F_Y represents the p-dimensional cumulative distribution function constructed via a students-t copula with Gaussian margins.
More precisely,

$$F_Y(x_1, ..., x_p) = T_v^R(\Phi_1(x_1), ..., \Phi_p(x_p)),$$

where T_v^R is the students-t copula with v degrees of freedom and dispersion matrix $R = \mathbb{I}_{p \times p}$.
We write

$$Y \sim T_\Phi(v, R)$$

Appendix: Copula Matlab/R-Code

```
rtcopula <- function(n,R,df){

  p <- ncol(R)
  T <- matrix(NA, n, p)

  for(i in 1:n){
    r <- rmvt(1,R,df)
    for(j in 1:p){
      term1 <- pt(r[j], df)
      term2 <- qnorm(term1)

      T[i,j] <- term2
    }
  }
  return(T)
}
```

For the full code, see program listing **12.6** in

Linear Models and Time-Series Analysis: Regression, ANOVA, ARMA and GARCH, First Edition. Marc S. Paolella.