UDC 004.8
DOI https://doi.org/10.32782/tnv-tech.2025.1.8

A RANDOM FOREST-BASED METHOD FOR DETECTING FEATURE DEPENDENCIES: A COMPARISON WITH PEARSON CORRELATION, MUTUAL INFORMATION, AND DISTANCE CORRELATION

Lytvyn A. A. *– Postgraduate Student, Private Higher Education Establishment "European University"*
ORCID ID: 0009-0009-5496-1417

In this work, we introduce a novel method for identifying dependent features in datasets that lack a target variable, a critical challenge in unsupervised learning. Understanding feature dependencies is essential in many machine learning applications, including dimensionality reduction, feature selection, and data preprocessing, where capturing both linear and nonlinear relationships among features is necessary. Traditional methods for dependency detection, such as Pearson correlation, mutual information, and the correlation of distances, have been widely used but often exhibit limitations, particularly when dealing with complex, high-dimensional data or non-linear dependencies.

Our approach addresses these challenges by leveraging a synthetic dataset generation technique. Specifically, we create synthetic features by sampling from the empirical distributions of the original features. This ensures that synthetic features are statistically independent while preserving the overall structure of the data. We then label the original dataset instances as 1 and the synthetic ones as 0, forming a binary classification problem. A Random Forest classifier is trained to distinguish between these two classes, and the feature importance scores obtained from the trained model provide insights into which features exhibit dependency. Features that contribute significantly to the classification task are identified as dependent, while those with lower importance scores are considered independent.

To evaluate the effectiveness of our method, we compare it against well-established dependency detection techniques. Pearson correlation primarily captures linear dependencies, while mutual information and correlation of distances can account for more complex relationships. Our experimental results demonstrate that the proposed approach outperforms these traditional methods by consistently identifying the correct set of dependent features across various tested scenarios. Moreover, our method exhibits greater robustness to noise, making it a reliable tool for unsupervised feature dependency detection in real-world datasets.

Key words: *Machine Learning, Feature Dependencies, Random Forest, Unsupervised Learning.*

Литвин А. А. Метод на основі випадкового лісу для виявлення залежностей ознак: порівняння з кореляцією Пірсона, взаємною інформацією та кореляцією відстаней

У цій роботі ми представляємо новий метод ідентифікації залежних ознак у наборах даних без цільової змінної, що є критичним завданням у навчанні без учителя. Розуміння залежностей між ознаками є важливим для багатьох застосувань машинного навчання, зокрема для зменшення розмірності, вибору ознак і передобробки даних, де необхідно враховувати як лінійні, так і нелінійні взаємозв'язки між ознаками. Традиційні методи виявлення залежностей, такі як коефіцієнт кореляції Пірсона, взаємна інформація та кореляція відстаней, широко використовуються, проте часто мають обмеження, особливо при роботі зі складними, багатовимірними даними або нелінійними залежностями.

Наш підхід вирішує ці проблеми за допомогою генерації синтетичного набору даних. Зокрема, ми створюємо синтетичні ознаки, виконуючи вибірку з емпіричних розподілів вихідних ознак. Це гарантує, що синтетичні ознаки є статистично незалежними, водночас зберігаючи загальну структуру даних. Далі ми позначаємо об'єкти вихідного набору даних як 1, а синтетичного – як 0, формуючи задачу бінарної класифікації. Для розрізнення цих двох класів ми навчаємо класифікатор на основі випадкового лісу (Random Forest), а отримані показники важливості ознак дають змогу визначити, які ознаки є залежними. Ознаки, що суттєво впливають на класифікацію, вважаються залежними, тоді як ті, що мають низькі значення важливості, вважаються незалежними.

Для оцінки ефективності нашого методу ми порівнюємо його з відомими техніками виявлення залежностей. Кореляція Пірсона переважно виявляє лінійні залежності, тоді як взаємна інформація та кореляція відстаней дозволяють враховувати більш складні взаємозв'язки. Наші експериментальні результати показують, що запропонований підхід перевершує традиційні методи, стабільно визначаючи правильний набір залежних ознак у різних тестових сценаріях. Крім того, наш метод демонструє вищу стійкість до шуму, що робить його надійним інструментом для виявлення залежностей між ознаками у задачах навчання без учителя.

Ключові слова: *машинне навчання, залежності між ознаками, випадковий ліс, навчання без учителя.*

Introduction. Data mining, a crucial area of machine learning, involves extracting useful patterns and knowledge from large datasets. In various domains, identifying dependent features among a noisy collection of independent features is essential for uncovering meaningful relationships within the data. Dependent features can reveal correlations, causal relationships, or shared behaviours that contribute to understanding the underlying structure of the dataset. However, in real-world data, features are often intermixed with noise, and distinguishing dependencies from independent variables becomes a complex task. This problem is not just a matter of academic interest; it has significant practical implications. Accurately identifying dependent features is crucial because it directly impacts the computational cost, the risk of overfitting, and the overall decision-making process in machine learning models.

Traditional statistical methods like Pearson correlation and mutual information have long been used to detect such dependencies, but they often struggle to capture nonlinear and complex relationships in data. As the scale and complexity of data grow, the ability to accurately identify dependent features becomes critical for tasks like feature selection, dimensionality reduction, and improving the performance of predictive models. Failure to do so can lead to inefficient algorithms, increased computational costs, and models that overfit to noise rather than capturing true underlying patterns.

Here is a more formal problem statement that is going to be studied in the paper. Given a dataset $D = \{X_1, X_2, \ldots, X_n\}$, where each X_i is a feature vector of observations, our goal is to identify the set of features that exhibit dependencies among each other. These dependencies may be linear or nonlinear, but we are not interested in defining or characterising the specific nature of the dependencies. Instead, the task is to provide a list of features that are not independent noise. The problem is framed as follows:

- **Input**: A set of features $D = \{X_1, X_2, \ldots, X_n\}$, where each X_i represents a feature in the dataset.
- **Objective**: Identify list of features that show dependency.
- **Output**: A set $F[D] \subseteq \{X_1, X_2, \ldots, X_n\}$, where each $X_i \in F[D]$ is determined to be dependent on at least one other feature in the dataset.

The specific goal is to distinguish these dependent features from features that are purely independent noise without further exploring the precise characteristics of the dependencies. In this paper, we specifically focus on cases where the number of dependent features is restricted to three. In further studies it potentially can be generalised for any unknown number of dependent features.

Random Forest and Feature Importance. This work relies heavily on the Random Forest algorithm, an ensemble method, particularly its feature importance measure, to identify dependent features. Ensemble methods are techniques that combine the predictions of multiple models to create a stronger overall model. The idea is that by aggregating the predictions of several models, the ensemble reduces errors and improves accuracy compared to individual models.

A decision tree is a supervised learning model used for classification and regression tasks, characterized by its tree-like structure. In the context of a binary classification problem, a decision tree aims to classify data points into one of two classes. The tree consists of internal nodes, leaf nodes, and branches. Internal nodes represent decisions or tests on an attribute, branches represent the outcome of the test, and leaf nodes represent the final class label.

Formally, a decision tree (namely CART algorithm defined in [1]) for binary classification can be defined as a recursive partitioning of the feature space. The process begins at the root node, which contains the entire dataset. At each internal node t, the data is split into two subsets based on a feature X_j and a threshold θ that best separates the data into the two classes, aiming to minimize impurity or maximize information gain. This splitting process continues recursively, creating child nodes, until a stopping condition is met (e.g., all data points in a node belong to the same class, or a maximum tree depth is reached).

The impurity $I(t)$ at a node can be measured using the Gini impurity, which is particularly useful for binary classification. The Gini impurity is defined as:

$$I(t) = 1 - \sum_{k=1}^{2} p_k^2$$

where p_k is the proportion of class k instances at node t, and there are two classes in the binary classification problem. The goal is to choose the feature and threshold that result in the largest reduction in impurity from the parent node to the child nodes.

One can find more details in the book by Breiman et al. [1].

Decision trees are popular in binary classification due to their simplicity, interpretability, and ability to handle both numerical and categorical data. However, they can be prone to overfitting, especially when the tree is deep and complex. Techniques such as pruning, setting a maximum depth, or using ensemble methods like Random Forests can help mitigate overfitting and improve generalization.

Random Forest, introduced by Breiman [2], is an ensemble learning method that constructs multiple decision trees and combines their outputs to improve classification performance. In the context of a binary classification problem, Random Forest aims to classify data points into one of two classes by aggregating the predictions of individual decision trees.

Formally, let $D = \{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$ be the training dataset, where X_i represents the feature vector and $y_i \in \{0,1\}$ is the class label. The Random Forest algorithm can be described as follows:

1. For $b = 1$ to B (number of trees in the forest):

a. Draw a bootstrap sample D_b from the training data D.

b. Grow a decision tree T_b on D_b by recursively repeating the following steps for each node:

i. Select a random subset of features m from the total p features.

ii. Choose the best feature and threshold to split the node based on a criterion (e.g., Gini impurity).

iii. Split the node into two child nodes.

iv. Repeat steps i-iii until a stopping condition is met (e.g., maximum depth or minimum number of samples per node).

2. Aggregate the predictions of all B trees to make the final prediction. For a new data point X, the Random Forest prediction $\hat{y}$ is given by:

$$\hat{y} = \text{mode}\left(\{T_b(X) : b = 1, 2, \ldots, B\}\right)$$

where $T_b(X)$ is the prediction of the b-th tree.

The randomness introduced by bootstrapping the data and selecting random subsets of features at each split helps to reduce overfitting and improve the model's generalization ability.

Random Forests are particularly effective in handling high-dimensional data and capturing complex interactions between features, making them a powerful tool for binary classification problems.

A key advantage of Random Forests is their ability to measure **feature importance**. In this paper, we rely on **Gini importance** (well described in [3]), a method for computing feature importance, as implemented in the scikit-learn library [4]. Gini importance evaluates the ability of each feature to split the data into distinct classes, and we use it to identify potentially dependent features.

Formal definition of Gini feature importance is well defined in [3]: importance of a variable X_j for predicting Y by adding up the weighted impurity decreases $p(t)\Delta i(s_t,t)$ for all nodes t where X_j is used, averaged over all trees ϕ_m (for $m=1,\ldots,M$) in the forest:

$$\mathrm{Imp}(X_j)=\frac{1}{M}\sum_{m=1}^{M}\sum_{t\in\phi_m}1(j_t=j)\left[p(t)\Delta i(s_t,t)\right]$$

where $p(t)$ is the proportion $\frac{N_t}{N}$ of samples reaching t and where j_t denotes the identifier of the variable used for splitting node t. When using the Gini index as the impurity function, this measure is known as the *Gini importance* or *Mean Decrease Gini*.

The success of Random Forests is evident in their extensive application across disciplines, from life sciences to finance, where they are used to extract valuable insights from large, noisy datasets. In particular, Touw et al. [5] highlight the efficacy of Random Forests in biological data mining, where they have become a preferred tool for managing complex, high-dimensional data. Their ability to handle nonlinear relationships and variable interactions makes them a powerful technique for feature importance analysis.

Proposed Method. In this paper, we propose a novel method for identifying dependent features in an unsupervised learning setting. The goal is to distinguish dependent features from independent noise without making assumptions about the specific nature of these dependencies. The core idea relies on generating a synthetic dataset that breaks any existing dependencies between features by ensuring they are sampled independently.

Given a dataset $D=\{X_1,X_2,\ldots,X_n\}$, where each X_i is a feature, we generate a synthetic dataset $S=\{S_1,S_2,\ldots,S_n\}$ where each S_i is drawn from the empirical distribution of the corresponding X_i: $S_i\sim P(X_i)$. S_i is constructed with replacement (i.e. each element of the original X_i has a chance to appear multiple times in the synthetic dataset). It is crusial to note that each S_i is independent of the others S_j for $j\neq i$ by design, as each S_i is constructed independently.

The synthetic dataset S is constructed to be the same size as the original dataset D, with the key distinction being that the features in S are independent by design. Dependencies, if present, exist only in the original dataset D.

To detect dependencies, we construct a new dataset Z where the original data points are labelled as 1, and the synthetic data points are labelled as 0. This dataset structure can be summarised as follows:

$$Z=\{(X_i,1)\mid X_i\in D\}\cup\{(S_i,0)\mid S_i\in S\},$$

where $S_i\sim P(X_i)$ and S_i is independent of S_j for each $j\neq i$.

A Random Forest classifier is trained on this dataset Z, with the goal of distinguishing between the original and synthetic data. The intuition is that features that help the classifier differentiate between D and S must contain dependencies that were broken in the synthetic dataset. The **feature importance** scores provided by the Random Forest classifier are used to rank the importance of each feature. Features with high importance scores are likely to be dependent on other features, while those with low importance are more likely to be independent noise.

In our proposed method, the performance of the Random Forest classifier is tuned by selecting optimal hyperparameters. Random Forest is a robust method with relatively few hyperparameters to tune, making the process efficient. Additionally, the accuracy metric used to evaluate the classifier's ability to distinguish between original and synthetic data is intuitive and straightforward. We will perform a grid search to explore different combinations of parameters, such as the number of trees, maximum depth, minimum samples per split and others. The optimised model with the highest accuracy will be used to generate feature importance scores, which will then be employed to identify dependent features.

We rely on **Gini importance** as implemented in 'scikit-learn' [4], which measures how much a feature reduces impurity at each split in the decision trees.

By this approach, we can identify and discriminate dependent features. This method is compared against traditional techniques such as Pearson correlation, mutual information, and correlation of distances to evaluate its performance in identifying complex feature dependencies.

Schema of the Proposed Method

The proposed method can be summarized in the following steps:

1. **Input Dataset Processing**
   - o Given input dataset D with features $X_1, X_2, \ldots, X_n$
   - o Each feature X_i represents a column of observations
2. **Synthetic Dataset Generation**
   - o Create synthetic dataset S by independently sampling with replacement from each feature's empirical distribution
   - o For each feature X_i, generate $S_i \sim P(X_i)$
   - o S_i is independent of all other S_j for $j \neq i$ by design, as each feature is sampled independently
3. **Combined Dataset Construction**
   - o Create dataset Z by combining original and synthetic data
   - o Label original data points as 1: $(X_i, 1)$
   - o Label synthetic data points as 0: $(S_i, 0)$
   - o Dataset Z is perfectly balanced by design with equal number of 0 and 1 labels
4. **Random Forest Training and Optimization**
   - o Perform grid search over hyperparameters
   - o Use accuracy as optimization metric (optimal due to balanced classes)
   - o Train Random Forest with optimal hyperparameters on dataset Z
5. **Feature Importance Analysis**
   - o Extract Gini importance scores from optimized Random Forest
   - o Rank features based on importance scores
   - o Higher scores indicate likely dependent features
   - o Lower scores suggest independent (noise) features

This schema illustrates how the method transforms the unsupervised problem of detecting feature dependencies into a supervised learning task, leveraging the Random Forest classifier's ability to identify important features that distinguish between the

original and synthetic datasets. And the balanced nature of the constructed dataset Z makes accuracy an ideal metric for hyperparameter optimization.

Competitor Methods

Pearson Correlation

Pearson correlation [6] is one of the methods we compare against in this paper. It is a statistical measure that quantifies the linear relationship between two continuous variables. The Pearson correlation coefficient, denoted by r, ranges from -1 to 1, where $r=1$ indicates a perfect positive linear relationship, $r=-1$ indicates a perfect negative linear relationship, and $r=0$ suggests no linear relationship between the variables. The formula for Pearson correlation is:

$$r=\frac{\sum\left(X_i-\overline{X}\right)\left(Y_i-\overline{Y}\right)}{\sqrt{\sum\left(X_i-\overline{X}\right)^2\sum\left(Y_i-\overline{Y}\right)^2}}$$

where X_i and Y_i represent the individual data points, and X and Y are their respective means. Pearson correlation is widely used due to its simplicity and ease of interpretation for linear relationships [6]. For this study, we use the implementation provided in the numpy library [7].

The main advantage of Pearson correlation is its straightforward calculation and its effectiveness in detecting linear dependencies between variables. However, its primary limitation is that it might miss nonlinear relationships, thus nonlinear relationships between features may go undetected, making it less effective for problems where the dependencies are more complex. Moreover, Pearson correlation assumes that the data is normally distributed and free of significant outliers, which can skew the results.

In our context, Pearson correlation is a useful baseline to detect linear dependencies between features. However, since we are concerned with capturing a broader range of relationships (including nonlinear and more complex dependencies), Pearson correlation alone may miss important feature interactions that are critical for distinguishing dependent features from independent noise.

It is important to note that we will use absolute values of the Pearson correlation coefficient as we don't care about the direction of the relationship, we only care about the existance of the relationship. Thus higher values of the absolute values of Pearson correlation coefficient indicate stronger relationships between features.

Mutual Information

Mutual information (MI) [8] is a key measure from information theory that quantifies the amount of information obtained about one random variable through another. It captures both linear and nonlinear dependencies between variables, making it more versatile than simpler measures like Pearson correlation. The mutual information $I(X;Y)$ between two variables X and Y is defined as:

$$I(X;Y)=H(X)+H(Y)-H(X,Y)$$

where $H(X)$ and $H(Y)$ are the entropies of the variables, and $H(X,Y)$ is their joint entropy. MI measures how much knowing one variable reduces uncertainty about the other, making it a robust tool for detecting dependencies, even in complex datasets. In the context of this paper, MI helps identify dependent features by quantifying the shared information between features, without assuming any specific linear relationships. The higher the MI value is, the more dependent the features presumed to be.

Mutual information is highly suited to detecting a broad range of dependencies, including nonlinear and conditional relationships, making it a powerful tool for identifying dependent features in unsupervised settings. However, MI's reliance on the choice of hyperparameters can introduce variability in the results. Despite these limitations, mutual information remains a robust method for feature dependency detection, complementing the other methods we explore in this paper.

For the theoretical foundation of mutual information, we reference Cover and Thomas [8], and the practical estimation follows the implementation described in Kraskov et al. [9] and provided by 'scikit-learn' [4].

Practical Estimation of Mutual Information

In practice, mutual information is estimated from sample data rather than theoretical distributions. This requires approximating the joint probability distributions of the features using techniques such as k-nearest neighbours (k-NN). The method implemented in 'scikit-learn' [4] for mutual information, based on work by Kraskov et al. [9], uses a nearest-neighbours approach to estimate MI from samples. The MI between two variables X and Y is estimated by considering the number of neighbouring points in the joint space of the variables, which provides an approximation of the shared information.

However, mutual information estimation through k-NN faces a hyperparameter selection issue–specifically, the choice of $n\_neighbors$, which controls how many neighbours are considered in the estimation process. Selecting too few neighbours may lead to underestimation, while selecting too many can smooth out dependencies, missing subtle relationships. For this study, we use the default settings provided by 'scikit-learn' to ensure consistency across simulations, but acknowledge that the choice of this parameter can significantly impact the results.

Correlation of Distances

Distance correlation, as discussed in [10], is a method for detecting both linear and nonlinear dependencies between variables. Unlike traditional correlation measures, it captures complex relationships by analyzing the distances between data points. This method is particularly effective at identifying nonlinear dependencies, which are often missed by simpler metrics like Pearson correlation [6].

Distance correlation is well-suited for identifying dependent features in unsupervised settings due to its ability to capture a wide range of dependencies without assuming a specific functional form. For theoretical insights, we refer to Székely et al. [10], and for practical implementation, we utilize the 'dcor' library [10].

Higher values of distance correlation coefficient indicate stronger relationships between features.

Practical Estimation of Distance Correlation

For practical estimation, we rely on the 'dcor' library, which implements distance correlation as described by Székely et al. [11]. The key parameter in this method is the exponent p, with a default value of 1, representing the exponent of the Euclidean distance. Adjusting this exponent can alter the method's sensitivity to local versus global dependencies. For our simulations, we use the default settings in the 'dcor' library [11], but recognize that changing this parameter can influence the detection of relationships.

Data Generation for Simulations

For our simulations, we generate a dataset with two types of features: independent noise features and dependent features. First, we create seven independent noise features, each sampled from a standard normal distribution with a mean of zero and a standard deviation of one. These features represent random noise with no dependencies.

Next, we generate three dependent features. The first two features, *local_trigger* and *slope*, are also sampled from a standard normal distribution and are independent of

each other. The third feature introduces dependency among the three. Specifically, when the value of *local_trigger* is less than zero, the third feature is sampled from a standard normal distribution independently. However, when *local_trigger* is greater than zero, the third feature is calculated as:

$$X = \text{slope} + \text{strength\_of\_noise} \times \text{random\_noise}$$

where *random_noise* is drawn from a normal distribution with a mean of zero and a standard deviation of one. Afterward, we normalise the third feature by subtracting its sample mean and dividing by its sample standard deviation, ensuring that it is scaled to a standard normal distribution. Please note that *local_trigger* and *slope* are generated once for the whole dataset and are the same for all samples in that dataset.

The total sample size for all features is 10000. We will repeat the simulation 100 times to study the distribution of calculated statistics and the performance of the proposed method and competitors. It is important to note that there is no hidden meaning or "magic" behind the specific numbers used in the data generation process (*seven* independent features, *three* dependent features, *10000* samples, *100* repetitions per noise level). These values are used as default starting points for this study. Future research could explore how the method behaves with different numbers of features, noise strength, and sample sizes.

Simulation Results

Let us briefly review the structure and purpose of our simulations. We generate random samples containing both dependent and independent features, with varying degrees of noise added to the dependent features. For each noise level, we generate multiple datasets to evaluate the effectiveness of different methods in terms of accuracy and statistical significance, as well as to study the distribution of the statistics produced by each method. For detailed simulation outputs and additional analysis, please refer to the appendix.

Accuracy Results

Table 1

Accuracy Results for Different Methods Across Noise Levels

Method	Metric	Noise=0.01	Noise=0.1	Noise=1.0	Noise=10.0
Distance Correlation	Accuracy	0.448	0.447	0.690	0.680
	Precision	0.496	0.496	0.690	0.680
	Recall	0.823	0.820	1.000	1.000
Mutual Information	Accuracy	0.414	0.424	0.647	0.408
	Precision	0.465	0.473	0.647	0.466
	Recall	0.790	0.803	1.000	0.767
Pearson Correlation	Accuracy	0.437	0.437	0.441	0.449
	Precision	0.489	0.489	0.491	0.499
	Recall	0.803	0.803	0.813	0.817
Proposed Method	Accuracy	1.000	1.000	1.000	0.563
	Precision	1.000	1.000	1.000	0.720
	Recall	1.000	1.000	1.000	0.720

The accuracy results in Table 1 demonstrate that competitor methods tend to label additional features as dependent, showing a tendency for false positives across all noise levels. In contrast, the proposed method achieved perfect accuracy results for each

degree of added noise, except for the highest noise one, where it still showed better *accuracy* results against competitors except for Distance Correlation.

Distribution Analysis

Table 2

Lognormal Distribution Test Results for Distance Correlation (Kolmogorov-Smirnov Test)

Noise Level	Statistic Type	Test Statistic	p-value	Type of aggregation
0.01	Dependent	0.066	0.759	Average
	Independent	0.052	0.933	Average
	Dependent	0.039	0.997	Maximum
	Independent	0.075	0.595	Maximum
	Dependent	0.083	0.476	Minimum
	Independent	0.055	0.907	Minimum
0.10	Dependent	0.064	0.776	Average
	Independent	0.061	0.828	Average
	Dependent	0.040	0.995	Maximum
	Independent	0.066	0.756	Maximum
	Dependent	0.056	0.889	Minimum
	Independent	0.050	0.950	Minimum
1.00	Dependent	0.057	0.885	Average
	Independent	0.065	0.762	Average
	Dependent	0.069	0.706	Maximum
	Independent	0.072	0.655	Maximum
	Dependent	0.064	0.788	Minimum
	Independent	0.052	0.938	Minimum
10.00	Dependent	0.066	0.746	Average
	Independent	0.055	0.905	Average
	Dependent	0.053	0.928	Maximum
	Independent	0.077	0.571	Maximum
	Dependent	0.064	0.788	Minimum
	Independent	0.042	0.992	Minimum

Table 3

Lognormal Distribution Test Results for Mutual Information (Kolmogorov-Smirnov Test)

Noise Level	Statistic Type	Test Statistic	p-value	Type of aggregation
1	2	3	4	5
0.01	Dependent	0.044	0.985	Average
	Independent	0.081	0.504	Average
	Dependent	0.077	0.566	Maximum
	Independent	0.077	0.562	Maximum
	Dependent	0.096	0.931	Minimum
	Independent	NA	NA	Minimum
0.10	Dependent	0.057	0.884	Average
	Independent	0.055	0.911	Average

Table 3 (Continued)

1	2	3	4	5
	Dependent	0.113	0.143	Maximum
	Independent	0.060	0.848	Maximum
	Dependent	0.121	0.780	Minimum
	Independent	NA	NA	Minimum
1.00	Dependent	0.062	0.819	Average
	Independent	0.057	0.878	Average
	Dependent	0.064	0.780	Maximum
	Independent	0.057	0.886	Maximum
	Dependent	0.154	0.202	Minimum
	Independent	NA	NA	Minimum
10.00	Dependent	0.058	0.867	Average
	Independent	0.081	0.499	Average
	Dependent	0.076	0.589	Maximum
	Independent	0.056	0.889	Maximum
	Dependent	0.176	0.227	Minimum
	Independent	NA	NA	Minimum

Table 4

Lognormal Distribution Test Results for Pearson Correlation (Kolmogorov-Smirnov Test)

Noise Level	Statistic Type	Test Statistic	p-value	Type of aggregation
0.01	Dependent	0.074	0.625	Average
	Independent	0.094	0.324	Average
	Dependent	0.057	0.877	Maximum
	Independent	0.053	0.925	Maximum
	Dependent	0.108	0.177	Minimum
	Independent	0.110	0.168	Minimum
0.10	Dependent	0.074	0.622	Average
	Independent	0.101	0.246	Average
	Dependent	0.058	0.865	Maximum
	Independent	0.064	0.787	Maximum
	Dependent	0.131	0.060	Minimum
	Independent	0.102	0.236	Minimum
1.00	Dependent	0.048	0.965	Average
	Independent	0.084	0.452	Average
	Dependent	0.065	0.772	Maximum
	Independent	0.059	0.860	Maximum
	Dependent	0.108	0.179	Minimum
	Independent	0.107	0.186	Minimum
10.00	Dependent	0.071	0.670	Average
	Independent	0.056	0.895	Average
	Dependent	0.042	0.992	Maximum
	Independent	0.050	0.951	Maximum
	Dependent	0.125	0.082	Minimum
	Independent	0.088	0.396	Minimum

Table 5

Lognormal Distribution Test Results for Proposed Method (Kolmogorov-Smirnov Test)

Noise Level	Statistic Type	Test Statistic	p-value	Type of aggregation
0.01	Dependent	0.092	0.348	Average
	Independent	0.103	0.224	Average
	Dependent	0.056	0.891	Maximum
	Independent	0.099	0.260	Maximum
	Dependent	0.092	0.351	Minimum
	Independent	0.100	0.253	Minimum
0.10	Dependent	0.100	0.258	Average
	Independent	0.100	0.257	Average
	Dependent	0.081	0.506	Maximum
	Independent	0.088	0.403	Maximum
	Dependent	0.078	0.550	Minimum
	Independent	0.114	0.139	Minimum
1.00	Dependent	0.274	<0.001*	Average
	Independent	0.301	<0.001*	Average
	Dependent	0.280	<0.001*	Maximum
	Independent	0.244	<0.001*	Maximum
	Dependent	0.166	0.007*	Minimum
	Independent	0.278	<0.001*	Minimum
10.00	Dependent	0.085	0.445	Average
	Independent	0.052	0.938	Average
	Dependent	0.052	0.935	Maximum
	Independent	0.074	0.614	Maximum
	Dependent	0.068	0.724	Minimum
	Independent	0.078	0.544	Minimum

** Indicates rejection of lognormal distribution hypothesis at 5% significance level*

The Kolmogorov-Smirnov test results in Tables 2, 3, 4, and 5 examine the distribution of aggregated statistics (minimum, maximum, and average) across 100 simulation runs for each method. The results confirm that these aggregated statistics generally follow a lognormal distribution at the 5% significance level for both the proposed method and competitors, with the notable exception of noise level 1.0. At this noise level, as indicated by asterisks in Table 5, all aggregation types (minimum, maximum, and average) reject the lognormal distribution hypothesis. It should be noted that some results for mutual information are marked as NA (Not Available) due to technical limitations in computing the statistics under certain conditions.

Mann-Whitney U Test Analysis

We conducted Mann-Whitney U tests (Table 6) to compare the minimum statistics values for dependent features against the maximum statistics values for independent features. This comparison is crucial because higher statistics values indicate stronger dependency signals between features. Therefore, optimal results would show a clear separation between the minimum dependent and maximum independent statistics values. We found that for competitor methods average *maximum independent statistics* were

higher than average *minimum dependent statistics* (column Dependent > Independent is equal to *False*) for all noise levels, which indicates that the distinction between noise and dependent features might not be clear. On the other hand, for the proposed method, the average *minimum dependent statistics* were higher than the average *maximum independent statistics* for all noise levels, except for noise level 10.00 (the largest noise level tested), and the difference is statistically significant (p-value = 0.001), indicating a clear separation between dependent and independent features.

Table 6

Mann-Whitney U Test Results

Method	Noise Level	Statistic Type	p-value	Dependent > Independent
Distance Correlation	0.01	max vs min	1.0	False
	0.10	max vs min	1.0	False
	1.00	max vs min	1.0	False
	10.00	max vs min	1.0	False
Mutual Information	0.01	max vs min	1.0	False
	0.10	max vs min	1.0	False
	1.00	max vs min	1.0	False
	10.00	max vs min	1.0	False
Pearson Correlation (abs)	0.01	max vs min	1.0	False
	0.10	max vs min	1.0	False
	1.00	max vs min	1.0	False
	10.00	max vs min	1.0	False
Proposed Method	0.01	max vs min	0.0*	True
	0.10	max vs min	0.0*	True
	1.00	max vs min	0.0*	True
	10.00	max vs min	0.999	False

** Indicates statistical significance at 5% level ($p < 0.05$)*

t-Test Analysis

We conducted t-tests on log-transformed statistics (Table 7) to compare the minimum statistics values for dependent features against the maximum statistics values for independent features, after confirming that the statistics follow lognormal distributions. Similar to the Mann-Whitney U test analysis, this comparison helps assess the separation between dependent and independent features. The results show that for competitor methods, the maximum independent statistics were higher than minimum dependent statistics across all noise levels, indicating poor separation. In contrast, the proposed method demonstrated clear separation with minimum dependent statistics being higher than maximum independent statistics for noise levels 0.01, 0.1, and 1.0, with statistical significance ($p < 0.001$). However, it's worth noting that for noise level 1.0, we could not confirm the lognormal distribution assumption for the proposed method's statistics, so these particular t-test results should be interpreted with caution as the normality assumption may not hold. At the highest noise level (10.0), the proposed method, like the competitors, failed to maintain separation between dependent and independent features.

Table 7

t-Test Results on Log-Transformed Statistics

Method	Noise Level	Statistic Type	p-value	Dep > Ind
Distance Correlation	0.01	max vs min	1.000	False
	0.10	max vs min	1.000	False
	1.00	max vs min	1.000	False
	10.00	max vs min	1.000	False
Mutual Information	0.01	max vs min	1.000	False
	0.10	max vs min	1.000	False
	1.00	max vs min	1.000	False
	10.00	max vs min	1.000	False
Pearson Correlation (abs)	0.01	max vs min	1.000	False
	0.10	max vs min	1.000	False
	1.00	max vs min	1.000	False
	10.00	max vs min	1.000	False
Proposed Method	0.01	max vs min	<0.001*	True
	0.10	max vs min	<0.001*	True
	1.00	max vs min	<0.001*	True
	10.00	max vs min	0.999	False

** Indicates statistical significance at 5% level ($p < 0.05$)*

Confidence Interval Analysis

Table 8

Confidence Intervals for Distance Correlation

Noise Level	Statistic Type	Lower Bound	Upper Bound
0.01	Max Independent	0.022	0.035
	Min Dependent	0.011	0.021
0.10	Max Independent	0.022	0.035
	Min Dependent	0.011	0.021
1.00	Max Independent	0.022	0.035
	Min Dependent	0.011	0.026
10.00	Max Independent	0.022	0.034
	Min Dependent	0.011	0.026

Table 9

Confidence Intervals for Mutual Information

Noise Level	Statistic Type	Lower Bound	Upper Bound
0.01	Max Independent	0.011	0.026
	Min Dependent	0.000	0.020
0.10	Max Independent	0.011	0.026
	Min Dependent	0.000	0.017
1.00	Max Independent	0.011	0.026
	Min Dependent	0.000	0.041
10.00	Max Independent	0.011	0.025
	Min Dependent	0.000	0.028

Table 10

Confidence Intervals for Pearson Correlation (absolute values)

Noise Level	Statistic Type	Lower Bound	Upper Bound
0.01	Max Independent	0.017	0.034
	Min Dependent	0.000	0.048
0.10	Max Independent	0.017	0.034
	Min Dependent	0.000	0.055
1.00	Max Independent	0.017	0.034
	Min Dependent	0.000	0.040
10.00	Max Independent	0.018	0.033
	Min Dependent	0.000	0.044

Table 11

Confidence Intervals for Proposed Method

Noise Level	Statistic Type	Lower Bound	Upper Bound
0.01	Max Independent	0.081	0.090
	Min Dependent	0.120	0.141
0.10	Max Independent	0.081	0.090
	Min Dependent	0.119	0.143
1.00	Max Independent	0.083	0.100
	Min Dependent	0.102	0.121
10.00	Max Independent	0.062	0.085
	Min Dependent	0.059	0.083

The confidence intervals presented in Tables 8, 9, 10, and 11 were constructed assuming lognormal distributions of the test statistics, as supported by our Kolmogorov-Smirnov test results. These intervals reveal important patterns in the discriminative power of each method. All competitor methods (Distance Correlation, Mutual Information, and Pearson Correlation) exhibit similar patterns of substantial overlap between the confidence intervals of maximum independent and minimum dependent statistics across all noise levels. This consistent overlap suggests that these traditional methods face inherent challenges in reliably distinguishing between dependent and independent features. In contrast, the proposed method demonstrates different behavior, with non-overlapping confidence intervals for noise levels up to 1.00. This clear separation begins to diminish only at the highest noise level (10.00), where the intervals start to overlap. The maintenance of distinct confidence intervals under varying noise conditions, compared to the consistent overlap seen in competitor methods, provides statistical evidence for the superior discriminative capability of our proposed method.

Conclusions. In this paper, we have presented a novel method for detecting feature dependencies based on random forest feature importance scores. Through simulations and comparative analysis with established methods like distance correlation, mutual information, and Pearson correlation, we have demonstrated several key findings:

First, our proposed method shows superior accuracy in detecting dependencies across different noise levels compared to traditional approaches. It only fell behind Distance Correlation at the highest tested noise level. While competitor methods tend to produce false positives, our method maintains perfect accuracy up to moderate noise levels and continues to outperform most other methods even under high noise conditions.

Second, the confidence interval analysis reveals that our method provides better statistical separation between dependent and independent features. Unlike traditional methods, which show substantial overlap in their confidence intervals, our approach maintains distinct intervals until extreme noise levels, providing more reliable discrimination between feature types.

Third, the robustness of our method shows its consistent performance across various noise levels, only showing degradation at the highest noise level (10.0). This stability is particularly valuable in real-world applications where the strength of dependencies may vary.

However, there are limitations and areas for future research. The method's performance degradation at very high noise levels suggests room for improvement in extreme conditions. Additionally, future work could explore the method's behavior with different types of dependencies, varying numbers of features, and different sample sizes.

In conclusion, our proposed method represents a significant advancement in feature dependency detection, offering improved accuracy and reliability compared to traditional approaches. These improvements make it a valuable tool for various applications in data analysis and feature selection, particularly in scenarios where complex dependencies need to be identified in the presence of noise.

REFERENCES:

1. Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees* (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9781315139470

2. Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

3. Louppe, G. (2015). Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*.https://arxiv.org/abs/1407.7502

4. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

5. Touw, W. G., Bayjanov, J. R., Overmars, L., Backus, L., Boekhorst, J., Wels, M., & van Hijum, S. A. F. T. (2013). Data mining in the life sciences with random forest: A walk in the park or lost in the jungle? *Briefings in Bioinformatics, 14*(3), 315–326. https://doi.org/10.1093/bib/bbs034

6. Rodgers, J. L., & Nicewander, W. A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician, 42*(1), 59–66. https://www.stat.berkeley.edu/~rabbee/correlation.pdf

7. Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature, 585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

8. Cover, T. M., & Thomas, J. A. (2005). *Elements of information theory* (pp. 13–55). John Wiley & Sons. https://www.cs.columbia.edu/~vh/courses/LexicalSemantics/Association/Cover&Thomas-Ch2.pdf

9. Kraskov, A., Stögbauer, H., & Grassberger, P. (2004). Estimating mutual information. *Physical Review E, 69*, 066138. https://doi.org/10.1103/PhysRevE.69.066138

10. Székely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *The Annals of Statistics, 35*(6), 2769–2794. https://doi.org/10.1214/009053607000000505

11. Ramos-Carreño, C., & Torrecilla, J. L. (2023). dcor: Distance correlation and energy statistics in Python. *SoftwareX, 22*, 101326. https://doi.org/10.1016/j.softx.2023.101326

APPENDIX:

1. Simulation Code. https://github.com/arteml12345/Random-Forest-Based-Method-for-Detecting-Feature-Dependencies
2. Whole Simulation Output. https://github.com/arteml12345/Random-Forest-Based-Method-for-Detecting-Feature-Dependencies/blob/main/full_simulation_output.html
3. Simulation Results, Raw Sample Output. https://github.com/arteml12345/Random-Forest-Based-Method-for-Detecting-Feature-Dependencies/blob/main/statistics.csv