Getting Started With DataSimilarity: Quantifying Similarity of Datasets and Multivariate Two- and k-Sample Testing

Marieke Stolte
TU Dortmund University

Luca Sauer
TU Dortmund University

Jörg Rahnenführer
TU Dortmund University

Andrea Bommert
TU Dortmund University

Abstract

Quantifying the similarity of two or more datasets is a common task in various applications of statistics and machine learning, including two- or k-sample testing and meta- or transfer learning. The **DataSimilarity** package contains a variety of methods for quantifying the similarity of datasets. The package includes 36 methods of which 14 are implemented for the first time in R. The remaining are wrapper functions for methods with already existing implementations that unify and simplify the various input and output formats of different methods and bundle the methods of many existing R packages in a single package. In this vignette, we show the basic workflow for using the package.

Keywords: dataset similarity, two-sample testing, multi-sample testing.

1. Introduction

The challenge of quantifying how similar two or more datasets are arises in various contexts where two or more datasets should be compared. This could be in the context of transferring results of a prediction model from one dataset to another, as well as for assessing how close simulated data is to a real-world dataset. The most common usage is for two- or k-sample testing. Formally, the two-sample problem is defined as the testing problem

$$H_0 : F_1 = F_2 \text{ vs. } H_1 : F_1 \neq F_2. \tag{1}$$

A two-sample test, therefore, can be used to check whether the underlying distributions of two datasets coincide. Analogously, the k-sample problem is defined as

$$H_0 : F_1 = F_2 = \cdots = F_k \text{ vs. } H_1 : \exists i \neq j \in \{1, \ldots, k\} : F_i \neq F_j,$$

for k distributions $F_1, \ldots, F_k$.

Many different methods are proposed in the literature for quantifying the similarity of two or more datasets, and most of these define a two- or k-sample test. In this package, a subset of these methods are implemented, which were selected as relevant from a literature review

(Stolte, Kappenberg, Rahnenführer, and Bommert 2024). For more details on the methods and their selection, see the 'Details' vignette. In the following, the basic steps for using the **DataSimilarity** package are explained using real-world example datasets with different characteristics with regard to the scale level, number of datasets, and presence of a target variable in each dataset.

2. Workflow

In the following, the typical workflow for working with the package is demonstrated.

There are two different use cases with different workflows.

a) We already know which method to apply to our dataset comparison at hand.

b) We have two datasets that we want to compare, but we do not have a specific method in mind.

In both cases, we first load the package:

```
R> library("DataSimilarity")
```

In case a), the workflow for using the package would be to find the corresponding function for the method and apply it to the data. The full list of methods can also be found in the 'Details' vignette as well as in the `method.table` dataset.

In case b), the package can also be used as a tool for finding an appropriate method. This depends on the dataset characteristics. Here, we distinguish between numeric and categorical data, the number of datasets (two or more than two), and whether or not the datasets include a target variable. We demonstrate how to find and apply a method for different types of datasets in the following. The general workflow for case b) can be summarized as follows:

1. Load the package.
2. Call `findSimilarityMethod()` to find an appropriate similarity method.
3. Call `DataSimilarity()` or use the function corresponding to the method found in 2. to apply the chosen method to the datasets at hand.

For the 2nd step, we present six important special cases in the following for datasets with different characteristics and demonstrate the package workflow in each of these special cases. For finding the appropriate methods in 2., there is a list of criteria (e.g. applicability to numeric or categorical data) which can guide our choice of an appropriate method. These were previously introduced by Stolte *et al.* (2024). The desired criteria can be passed to the `findSimilarityMethod()` by setting the corresponding arguments to `TRUE`. The function returns by default the function names for all implemented and suitable methods. By setting `only.names = FALSE`, the full information on which criteria the method fulfills can be retrieved.

2.1. Exactly two numeric datasets without target variables

The dataset `dhfr` (Sutherland and Weaver 2004) from the **caret** package (Kuhn and Max 2008) is a binary classification dataset (regarding Dihydrofolate Reductase inhibition) consisting of

325 compounds of which 203 are labeled as 'active' and 122 as 'inactive'. The variables are 228 molecular descriptors. As the active and inactive compounds should differ in their descriptors, we divide the dataset according to the first variable that indicates the activity status.

```
R> data(dhfr, package = "caret")
R> act <- dhfr[dhfr$Y == "active", -1]
R> inact <- dhfr[dhfr$Y == "inactive", -1]
```

For finding an appropriate method, we can use the function `findSimilarityMethod()`. We specify that we have two numeric datasets. As two datasets is already the default, we only need to specify `Numeric = TRUE`:

```
R> findSimilarityMethod(Numeric = TRUE)
```

```
 [1] "Bahr"           "BallDivergence" "BF"            
 [4] "BG"             "BG2"            "BMG"           
 [7] "C2ST"           "CCS"            "CF"            
[10] "Cramer"         "DiProPerm"      "DISCOB"        
[13] "DISCOF"         "DS"             "Energy"        
[16] "engineerMetric" "FR"             "FStest"        
[19] "GGRL"           "GPK"            "HMN"           
[22] "Jeffreys"       "KMD"            "LHZ"           
[25] "MMCM"           "MMD"            "MW"            
[28] "NKT"            "OTDD"           "Petrie"        
[31] "RItest"         "Rosenbaum"      "SC"            
[34] "SH"             "Wasserstein"    "YMRZL"         
```

We can also get more information if we set `only.names = FALSE`:

```
R> findSimilarityMethod(Numeric = TRUE, only.names = FALSE)
```

```
                                     Method Implementation
1               Baringhaus and Franz (2010)           Bahr
2                         Pan et al. (2018) BallDivergence
3               Baringhaus and Franz (2010)             BF
4                    Biau and Gyorfi (2005)             BG
5                   Biswas and Ghosh (2014)            BG2
6     Biswas, Mukhopadhyay and Ghosh (2014)            BMG
7          C2ST (Lopez-Paz and Oquab, 2017)           C2ST
8                  Chen, Chen and Su (2018)            CCS
10                 Chen and Friedman (2017)             CF
13 Cramer test (Baringhaus and Franz, 2004)         Cramer
14        DiProPerm test (Wei et al., 2016)      DiProPerm
15          DISCO (Rizzo and Székely, 2010)         DISCOB
16          DISCO (Rizzo and Székely, 2010)         DISCOF
17                       Deb and Sen (2021)             DS
18  Energy statistic (Zech and Aslan, 2003)         Energy
```

```
19           Engineer metric (Rachev, 1991) engineerMetric
20               Friedman and Rafsky (1979)             FR
22                Paul, De and Ghosh (2022)         FStest
23                      Ganti et al. (1999)           GGRL
24                GPK (Song and Chen, 2023)            GPK
25           Hediger, Michel and Näf (2021)            HMN
26                     Jeffrey's divergence       Jeffreys
27                KMD (Huang and Sen, 2023)            KMD
28                  Li, Hu and Zhang (2022)            LHZ
29                  Mukherjee et al. (2022)           MMCM
30               MMD (Gretton et al., 2009)            MMD
31             Mukhopadhyay and Wang (2020)             MW
32 Ntoutsi, Kalousis and Theodoridis (2008)            NKT
33            Alvarez-Melis and Fusi (2020)           OTDD
34                            Petrie (2016)         Petrie
35                Paul, De and Ghosh (2022)         RItest
36                         Rosenbaum (2005)      Rosenbaum
37                     Song and Chen (2022)             SC
38           Schilling (1986), Henze (1988)             SH
39                    q-Wasserstein metrics    Wasserstein
40                         Yu et al. (2007)          YMRZL
   Target.Inclusion   Numeric             Categorical
1       Unfulfilled Fulfilled             Unfulfilled
2       Unfulfilled Fulfilled             Unfulfilled
3       Unfulfilled Fulfilled             Unfulfilled
4       Unfulfilled Fulfilled             Unfulfilled
5       Unfulfilled Fulfilled             Unfulfilled
6       Unfulfilled Fulfilled             Unfulfilled
7       Unfulfilled Fulfilled Conditionally Fulfilled
8       Unfulfilled Fulfilled             Unfulfilled
10      Unfulfilled Fulfilled             Unfulfilled
13      Unfulfilled Fulfilled             Unfulfilled
14      Unfulfilled Fulfilled             Unfulfilled
15      Unfulfilled Fulfilled             Unfulfilled
16      Unfulfilled Fulfilled             Unfulfilled
17      Unfulfilled Fulfilled             Unfulfilled
18      Unfulfilled Fulfilled             Unfulfilled
19      Unfulfilled Fulfilled             Unfulfilled
20      Unfulfilled Fulfilled             Unfulfilled
22      Unfulfilled Fulfilled             Unfulfilled
23        Fulfilled Fulfilled               Fulfilled
24      Unfulfilled Fulfilled             Unfulfilled
25      Unfulfilled Fulfilled               Fulfilled
26      Unfulfilled Fulfilled             Unfulfilled
27      Unfulfilled Fulfilled             Unfulfilled
28      Unfulfilled Fulfilled             Unfulfilled
29      Unfulfilled Fulfilled               Fulfilled
```

```
30      Unfulfilled Fulfilled Conditionally Fulfilled
31      Unfulfilled Fulfilled             Unfulfilled
32        Fulfilled Fulfilled             Unfulfilled
33        Fulfilled Fulfilled               Fulfilled
34      Unfulfilled Fulfilled               Fulfilled
35      Unfulfilled Fulfilled             Unfulfilled
36      Unfulfilled Fulfilled             Unfulfilled
37      Unfulfilled Fulfilled             Unfulfilled
38      Unfulfilled Fulfilled             Unfulfilled
39      Unfulfilled Fulfilled             Unfulfilled
40      Unfulfilled Fulfilled               Fulfilled
      Unequal.Sample.Sizes              p.Larger.N Multiple.Samples
1                Fulfilled               Fulfilled      Unfulfilled
2                Fulfilled               Fulfilled        Fulfilled
3                Fulfilled               Fulfilled      Unfulfilled
4              Unfulfilled               Fulfilled      Unfulfilled
5                Fulfilled               Fulfilled      Unfulfilled
6                Fulfilled               Fulfilled      Unfulfilled
7                Fulfilled Conditionally Fulfilled        Fulfilled
8                Fulfilled               Fulfilled      Unfulfilled
10               Fulfilled               Fulfilled      Unfulfilled
13               Fulfilled               Fulfilled      Unfulfilled
14               Fulfilled               Fulfilled      Unfulfilled
15               Fulfilled               Fulfilled        Fulfilled
16               Fulfilled               Fulfilled        Fulfilled
17               Fulfilled               Fulfilled      Unfulfilled
18               Fulfilled               Fulfilled        Fulfilled
19               Fulfilled               Fulfilled      Unfulfilled
20               Fulfilled               Fulfilled      Unfulfilled
22               Fulfilled               Fulfilled        Fulfilled
23               Fulfilled               Fulfilled      Unfulfilled
24               Fulfilled               Fulfilled      Unfulfilled
25 Conditionally Fulfilled               Fulfilled      Unfulfilled
26               Fulfilled                    <NA>      Unfulfilled
27               Fulfilled               Fulfilled        Fulfilled
28               Fulfilled               Fulfilled      Unfulfilled
29               Fulfilled               Fulfilled        Fulfilled
30               Fulfilled               Fulfilled      Unfulfilled
31               Fulfilled               Fulfilled        Fulfilled
32               Fulfilled               Fulfilled      Unfulfilled
33               Fulfilled               Fulfilled      Unfulfilled
34               Fulfilled               Fulfilled        Fulfilled
35               Fulfilled               Fulfilled        Fulfilled
36               Fulfilled               Fulfilled      Unfulfilled
37               Fulfilled               Fulfilled        Fulfilled
38               Fulfilled               Fulfilled      Unfulfilled
39               Fulfilled               Fulfilled      Unfulfilled
```

```
40               Fulfilled               Fulfilled      Unfulfilled
          Without.training No.assumptions No.parameters Implemented
1                Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
2                Fulfilled      Fulfilled   Unfulfilled   Fulfilled
3                Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
4                Fulfilled      Fulfilled   Unfulfilled        <NA>
5                Fulfilled    Unfulfilled     Fulfilled        <NA>
6                Fulfilled    Unfulfilled     Fulfilled        <NA>
7              Unfulfilled    Unfulfilled   Unfulfilled   Fulfilled
8                Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
10               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
13               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
14               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
15               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
16               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
17               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
18               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
19               Fulfilled    Unfulfilled   Unfulfilled        <NA>
20               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
22               Fulfilled      Fulfilled   Unfulfilled   Fulfilled
23               Fulfilled      Fulfilled   Unfulfilled        <NA>
24               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
25 Conditionally Fulfilled      Fulfilled   Unfulfilled   Fulfilled
26               Fulfilled      Fulfilled     Fulfilled   Fulfilled
27               Fulfilled      Fulfilled   Unfulfilled   Fulfilled
28               Fulfilled      Fulfilled     Fulfilled        <NA>
29               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
30               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
31               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
32               Fulfilled      Fulfilled   Unfulfilled        <NA>
33               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
34               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
35               Fulfilled      Fulfilled   Unfulfilled   Fulfilled
36               Fulfilled    Unfulfilled     Fulfilled   Fulfilled
37               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
38               Fulfilled    Unfulfilled   Unfulfilled        <NA>
39               Fulfilled    Unfulfilled   Unfulfilled   Fulfilled
40             Unfulfilled      Fulfilled   Unfulfilled        <NA>
                       Complexity Interpretable.units Lower.bound
1                            <NA>         Unfulfilled           0
2                            <NA>         Unfulfilled           0
3                            <NA>         Unfulfilled           0
4                            <NA>         Unfulfilled           0
5                            <NA>         Unfulfilled           0
6                    O(N^2 log N)           Fulfilled           1
7                            <NA>           Fulfilled           0
8                            <NA>         Unfulfilled           0
```

```
10                           <NA>         Unfulfilled           0
13                         O(N^2)         Unfulfilled           0
14                           <NA>         Unfulfilled        <NA>
15                         O(N^2)         Unfulfilled           0
16                         O(N^2)         Unfulfilled           0
17                         O(N^3)         Unfulfilled           0
18                         O(N^2)         Unfulfilled           0
19                           <NA>         Unfulfilled           0
20                           <NA>           Fulfilled           2
22                           <NA>         Unfulfilled           0
23                           <NA>         Unfulfilled           0
24                           <NA>         Unfulfilled           0
25                           <NA>           Fulfilled           0
26                           <NA>         Unfulfilled           0
27                    O(KN log N)         Unfulfilled           0
28                           <NA>         Unfulfilled           0
29                           <NA>         Unfulfilled           0
30                  O(N^2p),O(Np)         Unfulfilled           0
31                           <NA>         Unfulfilled           0
32                           <NA>         Unfulfilled           0
33                           <NA>         Unfulfilled           0
34 O(N^2 log N),O(N^3),O(N log N)           Fulfilled   Fulfilled
35                           <NA>         Unfulfilled           0
36                         O(N^3)           Fulfilled           0
37                           <NA>         Unfulfilled           0
38                           <NA>           Fulfilled           0
39                           <NA>         Unfulfilled           0
40                           <NA>           Fulfilled           0
     Upper.bound      Rotation.invariant Location.change.invariant
1           <NA>               Fulfilled                 Fulfilled
2           <NA>                    <NA>                      <NA>
3           <NA>               Fulfilled                 Fulfilled
4              2             Unfulfilled               Unfulfilled
5    Unfulfilled               Fulfilled                 Fulfilled
6  min(n_1, n_2)               Fulfilled                 Fulfilled
7              1 Conditionally Fulfilled   Conditionally Fulfilled
8      Fulfilled               Fulfilled                 Fulfilled
10          <NA>               Fulfilled                 Fulfilled
13   Unfulfilled               Fulfilled                 Fulfilled
14          <NA> Conditionally Fulfilled   Conditionally Fulfilled
15   Unfulfilled               Fulfilled                 Fulfilled
16   Unfulfilled               Fulfilled                 Fulfilled
17          <NA>                    <NA>                 Fulfilled
18   Unfulfilled               Fulfilled                 Fulfilled
19   Unfulfilled             Unfulfilled                 Fulfilled
20             N               Fulfilled                 Fulfilled
22             1 Conditionally Fulfilled   Conditionally Fulfilled
```

```
23          <NA>             Unfulfilled                 Fulfilled
24          <NA> Conditionally Fulfilled   Conditionally Fulfilled
25             1             Unfulfilled                 Fulfilled
26   Unfulfilled                    <NA>                      <NA>
27             1               Fulfilled                 Fulfilled
28          <NA>                    <NA>                      <NA>
29          <NA>               Fulfilled                 Fulfilled
30          <NA> Conditionally Fulfilled   Conditionally Fulfilled
31          <NA>                    <NA>                      <NA>
32             1             Unfulfilled                 Fulfilled
33          <NA> Conditionally Fulfilled   Conditionally Fulfilled
34     Fulfilled               Fulfilled                 Fulfilled
35             1 Conditionally Fulfilled   Conditionally Fulfilled
36 min(n_1, n_2)               Fulfilled                 Fulfilled
37          <NA>               Fulfilled                 Fulfilled
38             1               Fulfilled                 Fulfilled
39          <NA> Conditionally Fulfilled   Conditionally Fulfilled
40             1             Unfulfilled                 Fulfilled
   Homogeneous.scale.invariant Positive.definite   Symmetric
1                  Unfulfilled         Fulfilled   Fulfilled
2                         <NA>         Fulfilled   Fulfilled
3                  Unfulfilled         Fulfilled   Fulfilled
4                  Unfulfilled         Fulfilled   Fulfilled
5                    Fulfilled              <NA>   Fulfilled
6                    Fulfilled       Unfulfilled   Fulfilled
7      Conditionally Fulfilled              <NA>   Fulfilled
8                    Fulfilled              <NA>   Fulfilled
10                   Fulfilled              <NA>   Fulfilled
13                 Unfulfilled         Fulfilled   Fulfilled
14     Conditionally Fulfilled              <NA> Unfulfilled
15                 Unfulfilled              <NA>   Fulfilled
16                 Unfulfilled              <NA>   Fulfilled
17                   Fulfilled         Fulfilled   Fulfilled
18                 Unfulfilled         Fulfilled   Fulfilled
19                 Unfulfilled       Unfulfilled   Fulfilled
20                   Fulfilled       Unfulfilled   Fulfilled
22                        <NA>              <NA>   Fulfilled
23                   Fulfilled              <NA>   Fulfilled
24     Conditionally Fulfilled              <NA>   Fulfilled
25                   Fulfilled              <NA>   Fulfilled
26                   Fulfilled         Fulfilled   Fulfilled
27                   Fulfilled         Fulfilled   Fulfilled
28                        <NA>         Fulfilled   Fulfilled
29                   Fulfilled              <NA>   Fulfilled
30     Conditionally Fulfilled         Fulfilled   Fulfilled
31                        <NA>              <NA>   Fulfilled
32                   Fulfilled       Unfulfilled   Fulfilled
```

```
33                        <NA>         Fulfilled   Fulfilled
34                   Fulfilled              <NA>   Fulfilled
35                        <NA>              <NA>   Fulfilled
36                   Fulfilled       Unfulfilled   Fulfilled
37                   Fulfilled              <NA>   Fulfilled
38                   Fulfilled       Unfulfilled   Fulfilled
39                 Unfulfilled         Fulfilled   Fulfilled
40                   Fulfilled              <NA>   Fulfilled
   Triangle.inequality           Consistency.N
1                 <NA>               Fulfilled
2          Unfulfilled               Fulfilled
3                 <NA>               Fulfilled
4                 <NA>               Fulfilled
5                 <NA>               Fulfilled
6                 <NA>                    <NA>
7                 <NA> Conditionally Fulfilled
8                 <NA>               Fulfilled
10                <NA>               Fulfilled
13           Fulfilled               Fulfilled
14                <NA> Conditionally Fulfilled
15                <NA>               Fulfilled
16                <NA>               Fulfilled
17                <NA>               Fulfilled
18           Fulfilled               Fulfilled
19           Fulfilled          Not Applicable
20                <NA>               Fulfilled
22                <NA>                    <NA>
23                <NA>                    <NA>
24                <NA>                    <NA>
25                <NA> Conditionally Fulfilled
26         Unfulfilled          Not Applicable
27                <NA>               Fulfilled
28                <NA>               Fulfilled
29                <NA>               Fulfilled
30           Fulfilled Conditionally Fulfilled
31                <NA>                    <NA>
32                <NA>          Not Applicable
33           Fulfilled          Not Applicable
34                <NA>                    <NA>
35                <NA>                    <NA>
36                <NA>               Fulfilled
37                <NA>               Fulfilled
38                <NA>               Fulfilled
39           Fulfilled          Not Applicable
40                <NA>                    <NA>
             Consistency.p Number.Fulfilled Number.Cond.Fulfilled
1                     <NA>               12                     0
```

```
2                     <NA>               11                     0
3                     <NA>               12                     0
4                     <NA>                9                     0
5                Fulfilled               13                     0
6                Fulfilled               13                     0
7                     <NA>                7                     6
8                     <NA>               13                     0
10                    <NA>               12                     0
13 Conditionally Fulfilled               14                     1
14                    <NA>                5                     5
15                    <NA>               11                     0
16                    <NA>               11                     0
17                    <NA>               13                     0
18 Conditionally Fulfilled               14                     1
19          Not Applicable                8                     0
20             Unfulfilled               14                     0
22               Fulfilled               11                     3
23                    <NA>               11                     0
24                    <NA>                8                     3
25                    <NA>               11                     3
26          Not Applicable               11                     0
27                    <NA>               16                     0
28                    <NA>               10                     0
29                    <NA>               14                     0
30                    <NA>                9                     5
31                    <NA>                9                     0
32          Not Applicable               11                     0
33          Not Applicable               11                     2
34                    <NA>               13                     0
35               Fulfilled               11                     3
36                    <NA>               14                     0
37                    <NA>               12                     0
38             Unfulfilled               12                     1
39          Not Applicable                9                     2
40                    <NA>               11                     0
   Number.Unfulfilled Number.NA
1                   6         3
2                   5         5
3                   6         3
4                   9         3
5                   5         3
6                   5         3
7                   5         3
8                   5         3
10                  5         4
13                  6         0
14                  6         5
```

```
15                  7         3
16                  7         3
17                  4         4
18                  6         0
19                 10         1
20                  6         1
22                  3         4
23                  4         6
24                  5         5
25                  4         3
26                  5         3
27                  3         2
28                  4         7
29                  3         4
30                  5         2
31                  4         8
32                  6         2
33                  4         2
34                  4         4
35                  3         4
36                  5         2
37                  5         4
38                  6         2
39                  7         1
40                  5         5
                                                     Class
1                Comparison based on inter-point distances
2                                         Testing approach
3                Comparison based on inter-point distances
4  Comparison of CDFs, density or characteristic functions
5                Comparison based on inter-point distances
6                                              Graph-based
7                    Method based on binary classification
8                                              Graph-based
10                                             Graph-based
13               Comparison based on inter-point distances
14                   Method based on binary classification
15               Comparison based on inter-point distances
16               Comparison based on inter-point distances
17               Comparison based on inter-point distances
18               Comparison based on inter-point distances
19                   Discrepancy measure for distributions
20                                             Graph-based
22                                        Testing approach
23 Comparison of CDFs, density or characteristic functions
24                                            Kernel-based
25                   Method based on binary classification
```

```
26                   Discrepancy measure for distributions
27                                            Kernel-based
28 Comparison of CDFs, density or characteristic functions
29                                             Graph-based
30                                            Kernel-based
31                                             Graph-based
32 Comparison of CDFs, density or characteristic functions
33               Distance/ similarity measure for datasets
34                                             Graph-based
35                                        Testing approach
36                                             Graph-based
37                                             Graph-based
38                                             Graph-based
39                   Discrepancy measure for distributions
40                   Method based on binary classification
                                    Subclass
1  Comparison based on inter-point distances
2                           Testing approach
3  Comparison based on inter-point distances
4                         Comparison of CDFs
5  Comparison based on inter-point distances
6                                Graph-based
7      Method based on binary classification
8                                Graph-based
10                               Graph-based
13 Comparison based on inter-point distances
14     Method based on binary classification
15 Comparison based on inter-point distances
16 Comparison based on inter-point distances
17 Comparison based on inter-point distances
18 Comparison based on inter-point distances
19                        Probability metric
20                               Graph-based
22                          Testing approach
23           Comparison of density functions
24                        Kernel-based (MMD)
25     Method based on binary classification
26                                Divergence
27                              Kernel-based
28    Comparison of characteristic functions
29                               Graph-based
30                        Kernel-based (MMD)
31                               Graph-based
32           Comparison of density functions
33 Distance/ similarity measure for datasets
34                               Graph-based
35                          Testing approach
```

```
36                                Graph-based
37                                Graph-based
38                           Graph-based (NN)
39                         Probability metric
40     Method based on binary classification
```

We could use this additional information and choose the method that fulfills most criteria among all methods that fulfill the required criteria, i.e., here, the KMD. For demonstration purposes, we apply the Rosenbaum cross-match test here to check whether the active and inactive compounds differ. For a description of the test, see the 'Details' vignette. As the combined sample size is smaller than 340, we can apply the exact test. We can either use the `DataSimilarity()` function and specify the `method` argument accordingly:

```
R> DataSimilarity(act, inact, method = "Rosenbaum", exact = TRUE)

        Exact cross-match test

data:  act and inact
z = -9.4098, p-value < 2.2e-16
alternative hypothesis: The distributions of act and inact are unequal.
sample estimates:
edge.count
        20
```

Alternatively, we can use the `Rosenbaum()` function directly:

```
R> Rosenbaum(act, inact, exact = TRUE)

        Exact cross-match test

data:  act and inact
z = -9.4098, p-value < 2.2e-16
alternative hypothesis: The distributions of act and inact are unequal.
sample estimates:
edge.count
        20
```

The output of the Rosenbaum test is an object of class '`htest`'. The output of the other methods is also in this format. The statistic value can be accessed by saving the result and accessing the `statistic` element of the saved result:

```
R> res.Rosenbaum <- Rosenbaum(act, inact, exact = TRUE)
R> res.Rosenbaum$statistic

        z
-9.409805
```

The *p* value can be accessed analogously as follows:

```
R> res.Rosenbaum$p.value

[1] 3.56166e-22
```

This holds for almost all other functions in this package. Additionally, the output might include more information specific to the method, which is then described on the respective help page. For the Rosenbaum test, for example, the unstandardized cross-match count is also returned and can be accessed via

```
R> res.Rosenbaum$estimate

edge.count
        20
```

The cross-match count is equal to 20. At most, there could be 122 cross-matches if each observation from the 'inactive' dataset was connected to an observation in the 'active' dataset. Therefore, the cross-match count of 20 can be considered a rather small value. This is also reflected by the z score of -9.41. Consequently, we see that the hypothesis of equal distributions can be rejected with a p value smaller than $2.2 \cdot 10^{-16}$.

We obtain a warning that informs us that a ghost value was introduced when calculating the optimal non-bipartite matching, due to the odd pooled sample size. This means that an artificial point was added to the sample that has the highest distance to all other points in the sample, such that the optimal non-bipartite matching, which needs an even sample size, could be calculated. The ghost value and the point with which it was matched are then discarded from the subsequent calculations.

2.2. More than two numeric datasets without target variables

The well-known `iris` dataset (Fisher 1936) included in the **datasets** package that comes with base R (R Core Team 2024) includes measurements of sepal and petals of 50 flowers each of three iris species. We compare the datasets for the three species Iris setosa, versicolor, and virginica, which are known to differ in their sepal and petal measurements.

```
R> data("iris")
R> setosa <- iris[iris$Species == "setosa", -5]
R> versicolor <- iris[iris$Species == "versicolor", -5]
R> virginica <- iris[iris$Species == "virginica", -5]
```

For finding an appropriate method, we can use the function `findSimilarityMethod()` again and specify that we have more than two numeric datasets using the `Numeric` and the `Multiple.samples` options:

```
R> findSimilarityMethod(Numeric = TRUE, Multiple.Samples = TRUE)

 [1] "BallDivergence" "C2ST"           "DISCOB"
 [4] "DISCOF"         "Energy"         "FStest"
 [7] "KMD"            "MMCM"           "MW"
[10] "Petrie"         "RItest"         "SC"
```

For comparing the three datasets, we could, for example, use the Mukherjee, Agarwal, Zhang, and Bhattacharya (2022) Mahalanobis multisample cross-match (MMCM) test, which is a generalization of the cross-match test for multiple samples. For a description of the test, see the 'Details' vignette. Again, we can either use the `DataSimilarity()` function or the `MMCM()` function directly

```
R> DataSimilarity(setosa, versicolor, virginica, method = "MMCM")

        Approximative MMCM test

data:  setosa, versicolor, virginica
chisq = 129.78, df = 3, p-value < 2.2e-16
alternative hypothesis: At least one pair of distributions are unequal.

R> MMCM(setosa, versicolor, virginica)

        Approximative MMCM test

data:  setosa, versicolor, virginica
chisq = 129.78, df = 3, p-value < 2.2e-16
alternative hypothesis: At least one pair of distributions are unequal.
```

The MMCM statistic value on its own is hard to interpret. However, the test rejects the null hypothesis of equal distributions with $p < 2.2 \cdot 10^{-16}$. Therefore, we can conclude that the observed MMCM value presents an extreme value when assuming the null. Thus, the datasets are dissimilar.

2.3. Exactly two numeric datasets with target variables

The `segmentationData` dataset (Hill, LaPan, Li, and Haney 2007) in the **caret** package (Kuhn and Max 2008) includes cell body segmentation data. The dataset contains 119 imaging measurements of 2019 cells to predict the segmentation that is divided into the two classes `PS` for 'poorly segmented' and `WS` for 'well segmented'. Moreover, there is a division into 1009 observations used for training and 1010 observations used as a test set. We compare this training and test set. Ideally, the distributions of the training and test set should be equal in this predictive modelling setting.

```
R> data(segmentationData, package = "caret")
R> test <- segmentationData[segmentationData$Case == "Test", -(1:2)]
R> train <- segmentationData[segmentationData$Case == "Train", -(1:2)]
```

The following methods would be appropriate to use:

```
R> findSimilarityMethod(Numeric = TRUE, Target.Inclusion = TRUE)

[1] "GGRL" "NKT"  "OTDD"
```

Setting `Target.Inclusion = TRUE` selects only the methods that can handle datasets that include a target variable. For demonstration, we choose the method of Ntoutsi, Kalousis, and Theodoridis (2008) and use all three proposed similarity measures NTO1, NTO2, and NTO3. For a description of the method, see the 'Details' vignette. The `target1` and `target2` arguments have to be specified as the column names of the target variable in the first and second supplied datasets, respectively. Here, the target variable is named `"Class"` in both cases. Again, we can use either the `DataSimilarity()` function or `NKT()`.

```
R> DataSimilarity(train, test, method = "NKT", target1 = "Class",
+                 target2 = "Class", tune = FALSE)

        Data similarity according to Ntoutsi et al. (2008), version 1

data:  train and test
s = 0.96931
alternative hypothesis: The distributions of train and test are unequal.

R> NKT(train, test, target1 = "Class", target2 = "Class", tune = FALSE)

        Data similarity according to Ntoutsi et al. (2008), version 1

data:  train and test
s = 0.96931
alternative hypothesis: The distributions of train and test are unequal.

R> DataSimilarity(train, test, method = "NKT", target1 = "Class",
+                 target2 = "Class", tune = FALSE, version = 2)

        Data similarity according to Ntoutsi et al. (2008), version 2

data:  train and test
s = 0.92444
alternative hypothesis: The distributions of train and test are unequal.

R> NKT(train, test, target1 = "Class", target2 = "Class", tune = FALSE,
+     version = 2)

        Data similarity according to Ntoutsi et al. (2008), version 2

data:  train and test
s = 0.92444
alternative hypothesis: The distributions of train and test are unequal.

R> DataSimilarity(train, test, method = "NKT", target1 = "Class",
+                 target2 = "Class", tune = FALSE, version = 3)
```

```
        Data similarity according to Ntoutsi et al. (2008), version 3

data:  train and test
s = 0.96648
alternative hypothesis: The distributions of train and test are unequal.

R> NKT(train, test, target1 = "Class", target2 = "Class", tune = FALSE,
+      version = 3)

        Data similarity according to Ntoutsi et al. (2008), version 3

data:  train and test
s = 0.96648
alternative hypothesis: The distributions of train and test are unequal.
```

We observe high similarity between the training and test datasets with all three methods, reflected by the similarity values `s` that are all close to the maximal value 1. For the method of Ntoutsi *et al.* (2008), no test is proposed and therefore, no p value is calculated.

2.4. Exactly two categorical datasets without target variables

The `banque` dataset from the **ade4** package (Dray and Dufour 2007) consists of bank survey data of 810 customers. All variables are categorical and contain socio-economic information of the customers. We divide the data into bank card owners and non-bank card owners and compare these two groups. In total, 243 out of the 810 customers own a bank card. Bank card owners and non-bank card owners might differ in their socio-economic characteristics.

```
R> data(banque , package = "ade4")
R> card <- banque[banque$cableue == "oui", -7]
R> no.card <- banque[banque$cableue == "non", -7]
```

We again apply the `findSimilarityMethod()` function to find appropriate methods for comparing two categorical datasets. Again, two samples are the default. Therefore, we only have to specify `Categorical = TRUE`.

```
R> findSimilarityMethod(Categorical = TRUE)

 [1] "C2ST"        "CCS_cat"     "CF_cat"      "CMDistance" "FR_cat"
 [6] "GGRL"        "HMN"         "MMCM"        "MMD"         "OTDD"
[11] "Petrie"      "YMRZL"       "ZC_cat"
```

For demonstration, we use the random forest test of Hediger, Michel, and Näf (2022) to compare these two groups. For a description of the test, see the 'Details' vignette. For easier interpretation, we look at the overall out-of-bag (OOB) prediction error instead of the per-class OOB prediction error and perform a permutation test with 1000 permutations. For reproducibility, we set a seed before applying the method. Alternatively, we could supply the seed via the `seed` argument for setting the seed within the function.

```
R> set.seed(1234)
R> DataSimilarity(card, no.card, method = "HMN", n.perm = 1000,
+                 statistic = "OverallOOB")

	Permutation OverallOOB random forest based two-sample test

data:  card and no.card
p.hat = 0.1605, p-value = 0.000999
alternative hypothesis: The distributions of card and no.card are unequal.

R> set.seed(1234)
R> HMN(card, no.card, n.perm = 1000, statistic = "OverallOOB")

	Permutation OverallOOB random forest based two-sample test

data:  card and no.card
p.hat = 0.1605, p-value = 0.000999
alternative hypothesis: The distributions of card and no.card are unequal.
```

The overall OOB prediction error is 0.161, which is considerably smaller than the naive prediction error of 243/810 = 0.3. Therefore, the random forest can distinguish between the datasets, so we can conclude that the datasets differ. This is also reflected by the *p* value of 9.990e-04.

2.5. More than two categorical datasets without target variables

We consider the banque dataset from the **ade4** package (Dray and Dufour 2007) again. This time, we split it by the nine socio-professional categories given by 'csp', which are again expected to differ with regard to the other socio-economic characteristics.

```
R> data(banque, package = "ade4")
R> agric <- banque[banque$csp == "agric", -1]
R> artis <- banque[banque$csp == "artis", -1]
R> cadsu <- banque[banque$csp == "cadsu", -1]
R> inter <- banque[banque$csp == "inter", -1]
R> emplo <- banque[banque$csp == "emplo", -1]
R> ouvri <- banque[banque$csp == "ouvri", -1]
R> retra <- banque[banque$csp == "retra", -1]
R> inact <- banque[banque$csp == "inact", -1]
R> etudi <- banque[banque$csp == "etudi", -1]
```

To compare these datasets, we now need a method that can handle multiple datasets at once:

```
R> findSimilarityMethod(Categorical = TRUE, Multiple.Samples = TRUE)

[1] "C2ST"   "MMCM"   "Petrie"
```

We apply the classifier two-sample test (C2ST). For a description of the test, see the 'Details' vignette. First, we use the default K-NN classifier. Categorical variables are dummy-coded. Again, we can use either `DataSimilarity()` or `C2ST()`:

```
R> DataSimilarity(agric, artis, cadsu, inter, emplo, ouvri, retra, inact,
+                 etudi, method = "C2ST")

        Approximative Classifier Two-Sample Test using knn

data:  agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi
p.hat = 0.31944, size = 567.00000, prob = 0.22593, p-value =
4.571e-07
alternative hypothesis: At least one pair of distributions are unequal.

R> C2ST(agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi)

        Approximative Classifier Two-Sample Test using knn

data:  agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi
p.hat = 0.31944, size = 567.00000, prob = 0.22593, p-value =
4.571e-07
alternative hypothesis: At least one pair of distributions are unequal.
```

The accuracy of the K-NN classifier is 0.319. It is larger than the naive accuracy for always predicting the largest class, which is given by `prob = 0.226` in the output. The classifier seems to be able to distinguish between the datasets, and we can therefore regard them as dissimilar. Moreover, the null hypothesis of equal distributions can be rejected with a p value of 4.571e-07.

For demonstration, we additionally perform the C2ST with a neural net classifier.

```
R> DataSimilarity(agric, artis, cadsu, inter, emplo, ouvri, retra, inact,
+                 etudi, method = "C2ST", classifier = "nnet",
+                 train.args = list(trace = FALSE))

        Approximative Classifier Two-Sample Test using nnet

data:  agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi
p.hat = 0.22222, size = 567.00000, prob = 0.22593, p-value =
0.001977
alternative hypothesis: At least one pair of distributions are unequal.

R> C2ST(agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi,
+       classifier = "nnet", train.args = list(trace = FALSE))

        Approximative Classifier Two-Sample Test using nnet
```

```
data:  agric, artis, cadsu, inter, emplo, ouvri, retra, inact, etudi
p.hat = 0.30556, size = 567.00000, prob = 0.22593, p-value =
1.826e-06
alternative hypothesis: At least one pair of distributions are unequal.
```

The results are very similar to using K-NN.

2.6. Exactly two categorical datasets with target variables

We consider the banque dataset from the **ade4** package (Dray and Dufour 2007) again. In this case, we interpret the savings bank amount (eparliv) variable as the target variable, which is again supplied via the target1 and target2 arguments. It is divided into the three categories '> 20000', '> 0 and < 20000', and 'nulle'. We divide the data into the socio-professional categories as before, and now need a method for two categorical datasets that include a target variable.

```
R> findSimilarityMethod(Categorical = TRUE, Target.Inclusion = TRUE)

[1] "GGRL" "OTDD"
```

We use the optimal transport dataset distance (OTDD) to compare the resulting datasets for craftsmen, shopkeepers, company directors ('artis'), to that of higher intellectual professions ('cadsu'), and to that of manual workers ('ouvri'). For a description of the method, see the 'Details' vignette. As all variables are categorical, we use the Hamming distance instead of the default Euclidean distance. We can either use DataSimilarity() or OTDD().

```
R> DataSimilarity(artis, cadsu, method = "OTDD", target1 = "eparliv",
+                 target2 = "eparliv", feature.cost = hammingDist)

        Optimal Transport Dataset Distance

data:  artis and cadsu
OTDD = 44.166
alternative hypothesis: Distributions of artis and cadsu are unequal

R> OTDD(artis, cadsu, target1 = "eparliv", target2 = "eparliv",
+       feature.cost = hammingDist)

        Optimal Transport Dataset Distance

data:  artis and cadsu
OTDD = 44.166
alternative hypothesis: Distributions of artis and cadsu are unequal
```

We obtain a dataset distance of 44.166 between craftsmen/shopkeepers/company directors and executives/higher intellectual professions. For the OTDD, low values correspond to high similarity, and the minimum value is 0. The observed value is clearly larger than zero, so the datasets are not exactly similar. How dissimilar they are is however hard to interpret from the observed OTDD value on its own. For the OTDD, no test is proposed and therefore, no p value is calculated.

```
R> DataSimilarity(artis, ouvri, method = "OTDD", target1 = "eparliv",
+                 target2 = "eparliv", feature.cost = hammingDist)

        Optimal Transport Dataset Distance

data:  artis and ouvri
OTDD = 49.427
alternative hypothesis: Distributions of artis and ouvri are unequal

R> OTDD(artis, ouvri, target1 = "eparliv", target2 = "eparliv",
+       feature.cost = hammingDist)

        Optimal Transport Dataset Distance

data:  artis and ouvri
OTDD = 49.427
alternative hypothesis: Distributions of artis and ouvri are unequal
```

We obtain a dataset distance of 49.427 between craftsmen/shopkeepers/company directors and manual workers. Again, this value on its own is hard to interpret. However, we can compare the values and conclude that the data of craftsmen/shopkeepers/company directors is more similar to that of executives/higher intellectual professions than to that of manual workers.

Acknowledgments

This work has been supported (in part) by the Research Training Group "Biostatistical Methods for High-Dimensional Data in Toxicology" (RTG 2624, Project P1) funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation - Project Number 427806116).
We would like to thank Nabarun Deb and Bodhisattva Sen for allowing us to use their R implementation of their test for the package. Moreover, we would like to thank David Alvarez-Melis, whose Python implementation of the OTDD was the basis for our R implementation.

References

Dray S, Dufour AB (2007). "The ade4 Package: Implementing the Duality Diagram for Ecologists." *Journal of Statistical Software*, **22**(4), 1–20. doi:10.18637/jss.v022.i04.

Fisher RA (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics*, **7**(2), 179–188. ISSN 2050-1439. doi:10.1111/j.1469-1809.1936.tb02137.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x.

Hediger S, Michel L, Näf J (2022). "On the Use of Random Forest for Two-Sample Testing." *Computational Statistics & Data Analysis*, **170**, 107435. ISSN 0167-9473. doi:10.1016/j.csda.2022.107435. URL https://www.sciencedirect.com/science/article/pii/S0167947322000159.

Hill AA, LaPan P, Li Y, Haney S (2007). "Impact of Image Segmentation on High-Content Screening Data Quality for SK-BR-3 Cells." *BMC Bioinformatics*, **8**(1), 340. ISSN 1471-2105. doi:10.1186/1471-2105-8-340. URL https://doi.org/10.1186/1471-2105-8-340.

Kuhn, Max (2008). "Building Predictive Models in R Using the caret Package." *Journal of Statistical Software*, **28**(5), 1–26. doi:10.18637/jss.v028.i05. URL https://www.jstatsoft.org/index.php/jss/article/view/v028i05.

Mukherjee S, Agarwal D, Zhang NR, Bhattacharya BB (2022). "Distribution-Free Multisample Tests Based on Optimal Matchings With Applications to Single Cell Genomics." *Journal of the American Statistical Association*, **117**(538), 627–638. ISSN 0162-1459. doi:10.1080/01621459.2020.1791131.

Ntoutsi I, Kalousis A, Theodoridis Y (2008). "A General Framework for Estimating Similarity of Datasets and Decision Trees: Exploring Semantic Similarity of Decision Trees." In *Proceedings of the 2008 SIAM International Conference on Data Mining (SDM)*, pp. 810–821. Society for Industrial and Applied Mathematics. ISBN 978-0-89871-654-2. doi:10.1137/1.9781611972788.73.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Stolte M, Kappenberg F, Rahnenführer J, Bommert A (2024). "Methods for Quantifying Dataset Similarity: A Review, Taxonomy and Comparison." *Statistics Surveys*, **18**, 163–298. ISSN 1935-7516. doi:10.1214/24-SS149.

Sutherland JJ, Weaver DF (2004). "Three-Dimensional Quantitative Structure-Activity and Structure-Selectivity Relationships of Dihydrofolate Reductase Inhibitors." *Journal of Computer-Aided Molecular Design*, **18**(5), 309–331. ISSN 0920-654X. doi:10.1023/b:jcam.0000047814.85293.da.

Affiliation:

Marieke Stolte
Department of Statistics
TU Dortmund University
Vogelpothsweg 87
44227 Dortmund, Germany
E-mail: stolte@statistik.tu-dortmund.de