Blessings of Dimensionality: Finding the optimal k for nearest-neighbor projected-distance feature selection (for detecting interaction and main effects in high dimensional data)

Bryan A. Dawkins[1], Trang T. Le[2] and Brett A. McKinney[1,3,*]
[1]Department of Mathematics, University of Tulsa, Tulsa, OK 74104, USA
[2]Department of Biostatistics, Epidemiology and Informatics, University of Pennsylvania, Philadelphia, PA 19104
[3]Tandy School of Computer Science, University of Tulsa, Tulsa, OK 74104, USA.
*Correspondence: brett.mckinney@gmail.com

Abstract

It is commonly known that high-throughput data has many inherent statistical challenges, such as multiple testing, sparsity and over fitting. Collectively, these challenges are known as the Curse of Dimensionality. Here we highlight an important Blessing of Dimensionality: the ability to identify interactions and main effects with neighborhoods of instances. We review the nearest-neighbor concept for finding interactions among attributes. We present a novel simulation method for generating data with both main effects and interactions from random networks with fine-tuned control over interaction effect size. Using our new simulation strategy, we determine optimal fixed k for neighborhood computation in nearest-neighbor distance-based feature selection for different combinations of data dimensions m (number of instances) and p (number of attributes), different ratios of main/interaction effect among functional attributes, and different combinations of main and interaction effect sizes. We discuss ways to maximize the blessings and minimize the curses of dimensionality to reliably identify interactions. Our results will show how optimal fixed k changes under a variety of conditions that we see in real data, which will serve as a guide for other researchers using nearest-neighbor distance-based feature selection.

Introduction

Relief-based methods identify interacting attributes as important by using nearest-neighbor information in higher dimensions (the "blessings of dimensionality"). Myopic methods that do not account for information from higher dimensions such as univariate tests are susceptible to false negatives when there are interactions. For example, in the plot of variable A versus C in a three-variable simulation (Fig. 1-I), variable A appears to show no difference between cases and controls (the marginal group means are the same). However, A is actually simulated to have a strong differential correlation with B, conditioned on the outcome variable (Fig. 1-II). Current Relief-based methods determine the importance of an attribute by computing the average difference of attribute A value between a target instance (X) and its nearest instance from the opposite class (Miss), $d_{\mathrm{X,M}}(A)$, subtracted from the similarly projected difference of target X and its nearest instance from the same class (Hit), $d_{\mathrm{X,H}}(A)$. A positive value from this calculation, i.e., $d_{\mathrm{X,M}}(A) - d_{\mathrm{X,H}}(A) > 0$, suggests that attribute A is useful for discriminating between cases and controls.

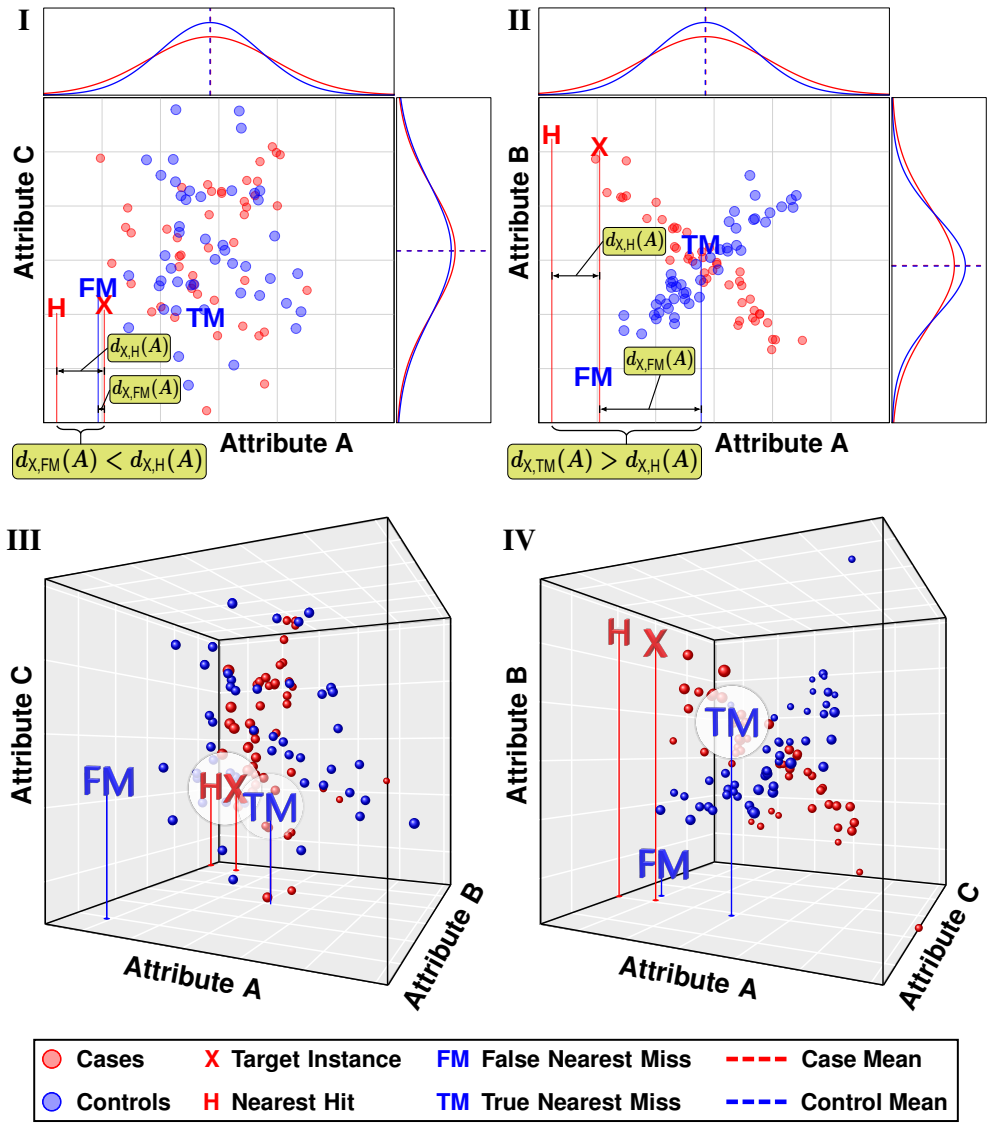

Fig 1. Imposters vs true neighbors in the presence of interactions with three variables. Attributes A, B, and C have no main effect. (**I**) Scatter plot of simulated irrelevant Attribute C with a functional Attribute A. (**II**) Scatter plot of Attributes A and B, which interact through differential correlation. Computing nearest neighbors with irrelevant attributes (**I**) or lower dimensions leads to imposter nearest neighbors and degrades the ability of Relief-based methods to identify interaction effects. Computing distances in only these two dimensions leads to an imposter false miss (FM) for the nearest neighbor from the opposite outcome class for target instance X. This imposter leads to Attribute A predicting closer projected distances for misses than hits (H), which incorrectly indicates that A is a poor discriminator (yellow boxes in **I**). (**III-IV**) Computing nearest neighbors in higher dimensions or with the correct interaction partner leads to imposter nearest neighbor (FM) being replaced by the true nearest miss neighbor (TM) for target instance X, which correctly leads to Attribute A predicting closer projected distances for hits (H) than misses, which is an indication that Attribute A is a good discriminator (yellow boxes **II**).

Relief-based methods use information from all available attributes (omnigenic) to estimate an attribute's importance. However, if relevant higher-dimensional information is not used to establish the neighborhoods of instances, these methods will miss the effect of A because "imposter" neighbors will be used in the attribute estimate (False Miss (FM) in Fig. 1-I, where $d_{\mathrm{X,FM}}(A) < d_{\mathrm{X,H}}(A)$). If one were to compute nearest neighbors in the A-C plane (ignoring the B dimension), the nearest miss would be an imposter (FM), which leads to a negative contribution to the importance score for A. One might call this C attribute a type-I confounding attribute because it increases the chances of interacting attributes to be false negatives. When nearest neighbors are calculated based on higher dimensions with relevant information (Fig. 1-III), it is clear that TM is closer to X than FM. The imposter (FM) is replaced by the true nearest miss (TM) and attribute A correctly shows a greater projected difference between misses than hits (Fig. 1-II $d_{\mathrm{X,TM}}(A) > d_{\mathrm{X,H}}(A)$), which is the signature of an important attribute. Univariate methods still cannot find the importance of A unless the interaction is explicitly modeled, but as long as functional variables A and B are in the space for nearest neighbor calculations (Fig. 1-III - IV), imposters can be excluded and Relief-based methods will find that A (and B) are important discriminators.

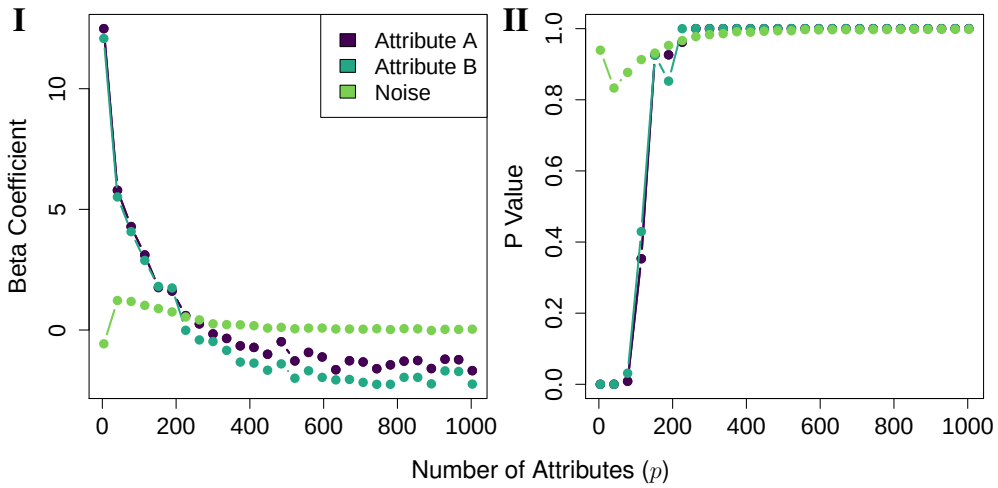

Fig 2. Interacting features A and B in the presence of noise. (**I**) Standardized beta coefficients of interacting features A and B and irrelevant features plotted vs the total number of features p. As more irrelevant features are added to the simulated interacting features A and B, beta coefficients decrease quickly. At about 200 features (198 irrelevant), A and B have beta coefficients close to 0. (**II**) P values corresponding to standardized beta coefficients of interacting features A and B and irrelevant features plotted vs the total number of features p. As more irrelevant features are added, P values quickly approach 1.

Using the same interaction partners (A and B), we explored the effects of increasing the level of noise (sparsity) in the data (Fig. 2). We iteratively added random null standard normal features and computed standardized beta coefficients (Fig. 2-I) and corresponding P values (Fig. 2-II) using NPDR [7]. For each value of p, we generated 10 separate sets of irrelevant attributes, combined them with attributes A and B, and computed beta coefficients and P values. After all 10 iterations for a fixed value of p, we took the average of the beta coefficients (Fig. 2-I) and the average of the corresponding P values (Fig. 2-II). As functionality becomes more sparse with respect to all features, it becomes increasingly difficult to detect interacting partners A and B. In fact, the beta

coefficients are approximately zero when there are approximately 200 features in this simulated data. By chance, there are many irrelevant attributes that have larger beta coefficients than A and B.

1 Neighborhood methods

Nearest-neighbor projected distance (NPD) methods rely on a neighborhood algorithm for feature selection. One may specify a fixed k number of neighbors, a fixed radius, an average radius SURF, a MultiSURF radius that adapts for each instance [1], or a variable-wise optimized k [2]. The definitive difference between fixed k and either fixed or adaptive radius methods is that there is always a deterministic number of neighbors in fixed k neighborhoods, but there is a variable neighborhood order in each fixed or adaptive radius method. In nearest-neighbor feature selection, neighborhood order is directly related to selected feature quality. Correlation and interaction effects change the distribution of pairwise distances between instances, which ultimately changes the probability of neighborhood inclusion for fixed or adaptive radius methods. In order to select the best set of features from data, it is important to know how pairwise feature correlation changes the optimal neighborhood order. We approach this problem by first determining the functional relationship between fixed k and MultiSURF radius in data with negligible pairwise feature correlation and no effects.

2 Derivation of expected k for MultiSURF neighborhoods

The MultiSURF radius for an instance is the mean of its distances to all other instances subtracted by $\alpha = 1/2$ of the standard deviation of this mean. Previously we showed empirically for balanced case-control datasets that a good constant k approximation to the expected number of neighbors within the multiSURF radii is $k = m/6$ [3], where m is the number of samples. Here we derive a more exact theoretical mean that shows the mathematical connection between neighbor-finding methods. This fixed k approximation to MultiSURF is independent of the type of data and the particular radii of each instance in the data.

The distance between instances ($i, j \in \mathcal{I}$, $|\mathcal{I}| = m$) in the data set $X^{m\times p}$ of m instances and p attributes is calculated in the space of all attributes ($a \in \mathcal{A}$, $|\mathcal{A}| = p$) using a metric such as

$$D_{ij}^{(q)} = \left(\sum_{a\in\mathcal{A}} |\mathrm{d}_{ij}(a)|^q \right)^{1/q}, \tag{1}$$

which is typically Manhattan ($q = 1$) but may also be Euclidean ($q = 2$). The quantity $\mathrm{d}_{ij}(a)$, known as a "diff" in Relief literature, is the projection of the distance between instances i and j onto the attribute a dimension. The function $\mathrm{d}_{ij}(a)$ supports any type of attributes (e.g., numeric and categorical). For example, the projected difference between two instances i and j for a continuous numeric ($\mathrm{d}^{\mathrm{num}}$) attribute a may be

$$\begin{aligned} \mathrm{d}_{ij}^{\mathrm{num}}(a) &= \mathrm{diff}(a, (i, j)) \\ &= |\hat{X}_{ia} - \hat{X}_{ja}|, \end{aligned} \tag{2}$$

where $\hat{X}$ represents the standardized data matrix X. We use a simplified $\mathrm{d}_{ij}(a)$ notation in place of the $\mathrm{diff}(a, (i, j))$ notation that is customary in Relief-based methods. We omit the division by $\max(a) - \min(a)$ used by Relief to constrain scores to the interval from -1 to 1. As we show in subsequent sections, NPDR scores are standardized

regression coefficients with corresponding P values, so any scaling operation at this stage is unnecessary for comparing attribute scores. The numeric $\mathrm{d}_{ij}^{\mathrm{num}}(a)$ projection is simply the absolute difference between row elements i and j of the data matrix $X^{m\times p}$ for the attribute column a.

We define the NPDR neighborhood set $\mathcal{N}$ of ordered pair indices as follows. Instance i is a point in p dimensions, and we designate the topological neighborhood of i as N_i. This neighborhood is a set of other instances trained on the data $X^{m\times p}$ and depends on the type of Relief neighborhood method (e.g., fixed k or adaptive radius) and the type of metric (e.g., Manhattan or Euclidean). If instance j is in the neighborhood of i ($j \in N_i$), then the ordered pair $(i,j) \in \mathcal{N}$ for the projected-distance regression analysis. The ordered pairs constituting the neighborhood can then be represented as nested sets:

$$\mathcal{N} = \{\{(i,j)\}_{i=1}^{m}\}_{\{j\neq i: j\in N_i\}}. \tag{3}$$

The cardinality of the set $\{j \neq i : j \in N_i\}$ is k_i, the number of nearest neighbors for subject i.

2.1 Predicted number of neighbors in the MultiSURF α neighborhood

Regardless of the predictor data type (numeric or categorical), the distribution of the p predictors (uniform, Gaussian, or binomial), or the metric used to compute distances (Manhattan or Euclidean), the $m(m-1)/2$ pairwise distances in the p-dimensional space are well approximated by a normal distribution. An instance j is in the adaptive α-radius neighborhood of i ($j \in N_i^{\alpha}$) under the condition

$$D_{ij} \leq R_i^{\alpha} \implies j \in N_i^{\alpha}, \tag{4}$$

where the threshold radius for instance i is

$$R_i^{\alpha} = \bar{D}_i - \alpha\, \sigma_{\bar{D}_i} \tag{5}$$

and

$$\bar{D}_i = \frac{1}{m-1}\sum_{j\neq i} D_{ij}^{(\cdot)} \tag{6}$$

is the average of instance i's pairwise distances (Eq. 1) with standard deviation $\sigma_{\bar{D}_i}$. MultiSURF implements $\alpha = 1/2$ [1].

The probability of the remaining $m-1$ instances being inside the α-radius of instance i (R_i^{α}) can be viewed as $m-1$ Bernoulli trials each with a probability of success q_{α}. Then the average average number of neighbors is given by

$$\bar{k}_{\alpha} = (m-1)q_{\alpha}, \tag{7}$$

from the mean of a binomial random variable. To calculate q_{α}, we assume the distribution of distances $\{D_{ij}\}_{j\neq i}$ of neighbors of instance i is normal $N(\bar{D}_i, \sigma_{\bar{D}_i})$. Our empirical studies confirm a normal distribution and that it is robust to data type and metric. Extreme violations of independence of attributes (extreme correlations or interactions) will cause the distribution to be right skewed, but this effect is difficult to observe in real data. Thus, for a Gaussian pairwise distance distribution, the probability q_{α} for one instance $j \neq i$ to be in the neighborhood of i ($j \in N_i^{\alpha}$) is given by the area under the mean-centered ($\bar{D}_i$) Gaussian from $-\infty$ to R_i^{α}. An illustration of the area computed to estimate q_{α} is given by Fig. 3. This integral can be written in terms of the error function (erf):

$$q_{\alpha} = \frac{1}{2}\left(1 - \mathrm{erf}\left(\frac{\alpha}{\sqrt{2}}\right)\right). \tag{8}$$

And finally using Eqs. (7 and 8) we find

$$\bar{k}_\alpha = \left\lfloor \frac{m-1}{2}\left(1 - \mathrm{erf}\left(\frac{\alpha}{\sqrt{2}}\right)\right)\right\rfloor, \tag{9}$$

where we apply the floor function to ensure the number of neighbors is integer. For data with balanced hits and misses in standard fixed k Relief, one divides this formula by 2. For MultiSURF ($\alpha = 1/2$), this formula gives $\bar{k}_\alpha^{\mathrm{hit/miss}} = \bar{k}_{1/2}^{\mathrm{hit/miss}} = \frac{1}{2}\bar{k}_{1/2} = .154(m-1)$, which is very close to our previous empirical estimate $m/6$. When we compare MultiSURF neighborhood methods with fixed k neighborhoods, we use $\bar{k}_{1/2}$. Using this $\alpha = 1/2$ value has been shown to give good performance for simulated data sets. However, the best value for α is likely data-specific and should probably be determined for each individual data set.

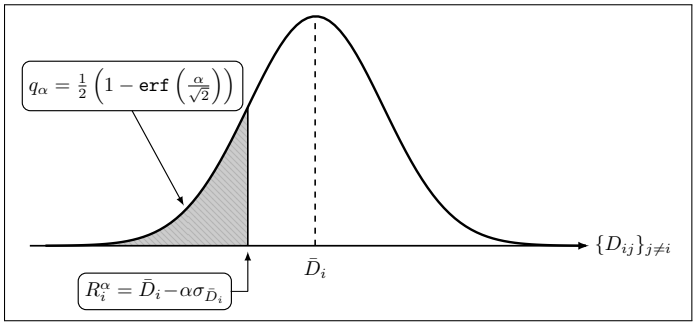

Fig 3. Illustration of the expected probability of a fixed instance j being in the fixed radius neighborhood of another instance i. The fixed radius is parameterized by a fraction α of the standard deviation of all pairwise distances measured from instance i to all possible neighbors.

3 Data simulations

In this section, we discuss our methods of generating interaction effects, main effects, and a combination of main and interaction effects. In our analysis of optimizing k, we use mixed simulations containing main and interaction effects to determine the best overall strategy of choosing k for a given scenario on the data.

3.1 Interaction effects

We extend an interaction effect simulation method that generates differential correlation starting from a random graph from either Erdős-Rényi or Scale-free degree distribution [4]. With this method, functional features in the control group are given large pairwise correlations with all other features. Differential correlations (interaction effects) are created by randomly permuting functional feature data entries within the case group only, which destroys the correlation structure in the case group but preserves the correlation structure in the control group. This method creates large effect sizes, which are easily detected by nearest-neighbor distance based methods. The reason for this ease of detection is the uniformity in low and high correlations in case and control groups, respectively.

In order to establish more influence over the number of differential pairwise correlations, we simulate correlation matrices for case and control groups directly. We allowed only functional connections to be given differential correlation between case and control groups, where a functional connection is simply the presence of an edge (or link) from one feature to a functional feature in the random network that is generated. We show a flow diagram that displays our simulation method for interactions (Figure 4). Analogous to the former method [4], we start with a random graph with either Erdös-Rényi or Scale-free degree distribution (Figure 4**(1)**). From the random graph, we choose

which attributes will be given functional interactions (F) among those with non-zero degree (Figure 4**(2)**). Using adjacency matrix A, along with the indices of interaction attributes F, we create correlation matrices for cases and controls (Figure 4**(3)**). For the control group, we assign high correlations (ρ^{hi}) to connected attribute pairs from the random graph while non-connected pairs are given low correlation (ρ^{lo}). We add noise ($\varepsilon_{ij} \sim \mathcal{N}(0, 0.1)$) to the high and low correlation parameters as well, which gives each pair of attributes a different correlation. The case group starts out with the same exact correlation matrix as the controls (P^{ctrl}). For attributes that are connected to another attribute that is functional (F), we create differential correlation for these pairs using the parameter

$$b^{\text{int}} = -t\rho^{\text{hi}} + (1-t)\rho^{\text{hi}}, \text{ where } t \in [0, 1]. \tag{10}$$

As $t \to 0$, the effect size decreases monotonically. On the other hand, the effect size increases monotonically as $t \to 1$. By controlling ρ^{hi}, b^{int}, and the level of network connectivity, we have the ability to more finely control the interaction effect size than the preceding simulation method upon which ours is drawn [4].

After correlation matrices are created for cases (P^{case}) and controls (P^{ctrl}), we compute the upper triangular Cholesky factors (U^{case} and U^{ctrl}) for each correlation matrix (Figure 4**(4)**). We then simulate null data matrices X^{case} and X^{ctrl} (Figure 4**(5)**) for cases and controls, respectively, such that

$$x_{ij}^{\text{case}}, x_{ij}^{\text{ctrl}} \sim \mathcal{N}(0, 1) \quad \forall i, j. \tag{11}$$

Multiplication of X^{case} and X^{ctrl} by the Cholesky factors U^{case} and U^{ctrl}, respectively, produce case and control sub-matrices Y^{case} and Y^{ctrl} with the correlation structure described previously. These sub-matrices are then combined into a single data $m \times p$ matrix given by

$$X^{\text{int}} = \begin{bmatrix} Y^{\text{ctrl}} \\ \text{- - - -} \\ Y^{\text{case}} \end{bmatrix}. \tag{12}$$

The final interaction data set (X^{int}) contains both functional interacting attributes and noise attributes with no effect. We can use the interaction simulation algorithm (Figure 4**(1)(5)**) to generate only interactions, or we can use the final step of the algorithm (Figure 4**(5)**) in combination with main effect simulations to create data with mixed effects (Figure 4**(6)**-**(7)**). We explain the main effect simulations (Figure 4**(6)**) in the next section.

3.2 Main effects

We use a method for generating main effects that was previously applied to the study of gene expression data [5]. This method uses a linear model given by

$$X_{ia} = b_a y_i + \varepsilon_{ia}, \tag{13}$$

where $b_a \sim \mathcal{N}(0, b^{\text{main}})$ for some fixed $b^{\text{main}} > 0$, $y_i \in \{0(\text{control}), 1(\text{case})\}$, and $\varepsilon_{ia} \sim \mathcal{N}(0, 1)$. We show an illustration of this method (Figure 4**(6)**), where the end result (X^{main}) contains three main effect attributes with effects sizes determined by $B^{\text{T}} = [b_1, b_2, b_3]$. The effect sizes vanish in the lower block of matrix $(BY)^{\text{T}}$ because this lower block represents the control group whose class labels are 0. The magnitude of $b^{\text{main}} > 0$ determines the effect size between case and control groups. We add noise (E) so that case and control groups are both normally distributed with different means with respect to each main effect, but have approximately the equal variance.

3.3 Mixed effects: interactions and main effects

In order to simulate data with both interactions and main effects, we combine the interaction simulations (Figure 4**(5)**) and main effect simulations (Figure 4**(6)**) into a single data set (X) (Figure 4**(7)**), where the left block matrix of X contains interactions and noise and the right block of X contains only main effects.

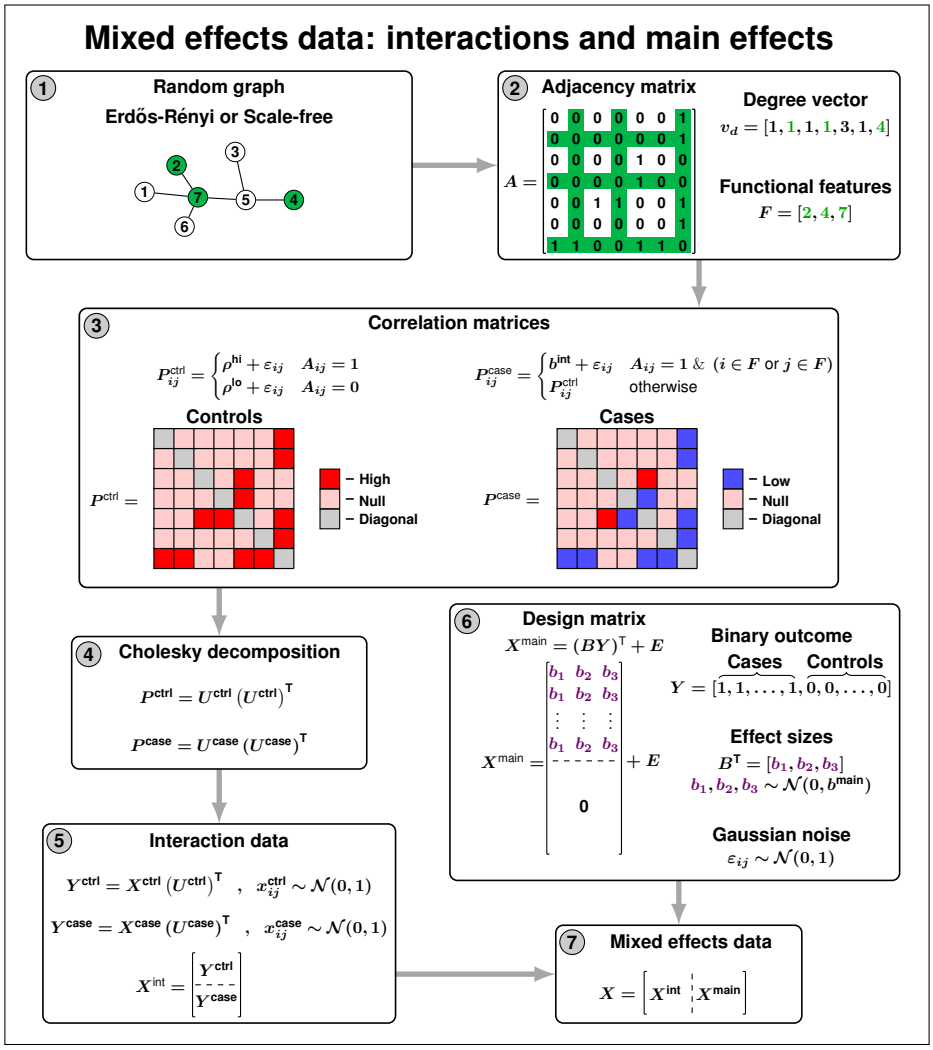

Fig 4. Algorithm for simulating mixed effects data with interactions and main effects. **(1)** Random network is generated, whose degree distribution is either Erdős-Rényi or Scale-free. **(2)** Adjacency matrix (A) and degree vector (v_d) corresponding to the random network are computed and functional features (F) are randomly selected from those with positive degree. **(3)** Correlation matrices are generated for cases and controls. In the control group, high (ρ^{hi}) and low (ρ^{lo}) correlations are assigned to connected ($A_{ij} = 1$) and non-connected ($A_{ij} = 0$) feature pairs, respectively. In the case group, differential correlation (b^{int}) is applied to functional connections. **(4)** Upper triangular Cholesky factors are computed for case/control correlation matrices. **(5)** Standard normal random data matrices (X^{ctrl} and X^{case}) are given correlation structure associated with case and control groups and combined into full data matrix with interaction effects (X). **(6)** Main effects simulated with effect sizes randomly sampled from $\mathcal{N}(0, b^{\text{main}})$. **(8)** Interactions ($X^{\text{int}}$) and main effects ($X^{\text{main}}$) combined X.

4 Optimizing k for detecting effects

All simulation scripts for interactions, main effects, and mixed effects, were written in **R** [6], and are available on Github (`https://github.com/insilico/npdr`). We generated simulation replicates for many different scenarios on the data. Each simulated data set $X^{(m \times p)}$ generated contained m instances and p features, where $m \in \{100, 250, 500\}$ and $p \in \{1000, 2500, 5000\}$. All combinations of m and p were explored in order to determine how neighborhood selection parameters change with dimensionality. We considered a balanced binary outcome only so that there were exactly $m/2$ cases and $m/2$ controls. In each simulation, 10% of the total number of features p were functionally related to the outcome variable while the remaining 90% were simply background features with no effect. Along with exploring all combinations of m and p, we also considered different ratios of main-to-interaction effect among functional attributes. We considered ratios of 0.1, 0.2, 0.3, ..., 0.9 of main-to-interaction, where 0.1 means that 10% of the functional attributes are main effects while 90% are interactions. For each simulation, we kept the main effect size parameter (b^{main}) fixed at 0.5 to reduce the total number of simulations. We explored different interaction effect sizes by varying the high correlation parameter (ρ^{hi}) only. The effect size increases monotonically with increasing ρ^{hi} because b^{int} is simply a point on the line from $-\rho^{\text{hi}}$ to ρ^{hi}, which is controlled by an input t in our simulation **R** function. We considered values of $\rho^{\text{hi}} \in \{0.3, 0.6, 0.9\}$ so that we can explore small, medium, and large interaction effect sizes across all scenarios. The non-connection, low correlation parameter (ρ^{lo}) was fixed at 0.1 for all simulations. Taking all of the simulation parameters into account, we have a total of $3 \times 3 \times 9 \times 3 = 243$ different scenarios on data within our simulated data sets. For each scenario, we generate 30 replicates so that we can see the average value of fixed k that optimizes functional attribute scores.

We use a method that we call Variable-Wise Optimized k (VWOK), formerly referred to as Gene-wise Adaptive k (GWAK) [2], to determine the optimal k for each attribute in a particular simulated data set (Figure 5). VWOK maximizes the scoring metric for each individual attribute, where we use standardized NPDR beta coefficients for scoring attributes in our analysis. The winning score is the one that maximizes the NPDR beta coefficient for each attribute. Winning scores are sorted from largest to smallest, along with the corresponding pseudo P values from NPDR, and those with significant adjusted P values are selected for further analysis. The VWOK method can be applied similarly using any nearest-neighbor feature selection algorithm, such as Relief-Based Algorithms [1]. The success of VWOK relies on the fact that scores of noise attributes do not change significantly as a function of k, while functional attribute scores do change significantly and have a well defined maximum with respect to k. Another reason we use VWOK is because we can explore all values of k, from $k = 1$ to $k = m - 1$, for each attribute.

The ability to detect main effects increases monotonically with increasing k, while detection of interactions is optimized for intermediate values of k. To explore this, we generated 30 data sets with main effects and 30 data sets with interactions, where each data set had $m = p = 100$ instances and attributes. For each data set and each value of k, we computed standardized NPDR beta coefficients. We then computed the area under the Precision-Recall Curve (auPRC) for each set of beta coefficients. For each value of k, we plotted the average auPRC from the 30 simulation replicates for both main effects and interaction effects (Figure 6). We observed that auPRC for data sets with main effects increases monotonically with increasing k. For data with interaction effects, there is a more complicated relationship between auPRC and k. We see an increase in auPRC between $k = 1$ and $k = 18$, but then there is no significant change until after $k = 38$. Between $k = 39$ and $k = 99$, auPRC for data with interactions decreases monotonically with increasing k. Such different detectability between interactions and main effects

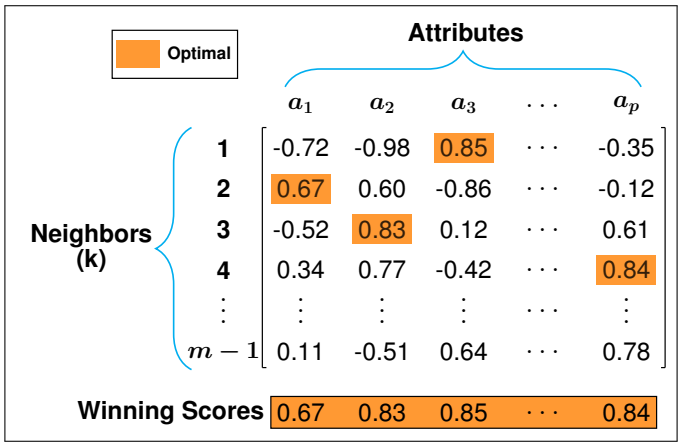

Fig 5. Variable-Wise Optimized k (VWOK) method for choosing optimal values of k in nearest-neighbor feature selection. For each attribute a and each value of $k = 1, 2, \ldots, m-1$, a score or weight is computed with a nearest-neighbor feature selection method like Relief-Based Algorithms (RBAs) or another similar method. Each feature is assigned the value of k that maximizes its importance score. Winning scores are then sorted in decreasing order and some fraction of the top scoring attributes are chosen for further analysis. In our analysis, we use standardized NPDR beta coefficients to score attributes, where adjusted pseudo P values are used to filter out insignificant attributes.

produces a dilemma for fixed k and fixed radius methods. If neighborhood selection parameters are too large, then neighborhoods will be tuned for detecting main effects but too large to detect interactions. A similar result happens if neighborhoods are too small, except we lose the ability to detect both types of effects. Real data typically has both interactions and main effects, so we must consider the implications of a particular parameterization for nearest-neighbor feature selection algorithms. With VWOK, we can tune k for each individual attribute to avoid having to choose between the different types of effects.

Because our goal in this analysis is not necessarily model selection, but rather feature selection where we know which features are functionally related to the outcome, we apply VWOK without splitting the data into training and test folds. Generally, one would intuitively apply a cross-validation method for parameter tuning to avoid overfitting. However, we have seen empirically that cross-validation does not produce high quality attributes in the context of nearest-neighbor feature selection. One reason that cross-validation does not perform well in nearest-neighbor feature selection algorithms is because it optimizes classification accuracy and does not optimize attribute scores. With nearest-neighbor feature selection, we have seen that cross-validation methods tend to produce highly predictive models composed of largely irrelevant attributes. Also, we have the advantage of knowing which attributes are functional in simulated data so our only concern is to determine which value of k maximizes the scores of these attributes. Because noise attribute scores do not change significantly as a function of k but functional attribute scores do, we maximize the probability of selecting these functional attributes among those at the top of the ranking with VWOK.

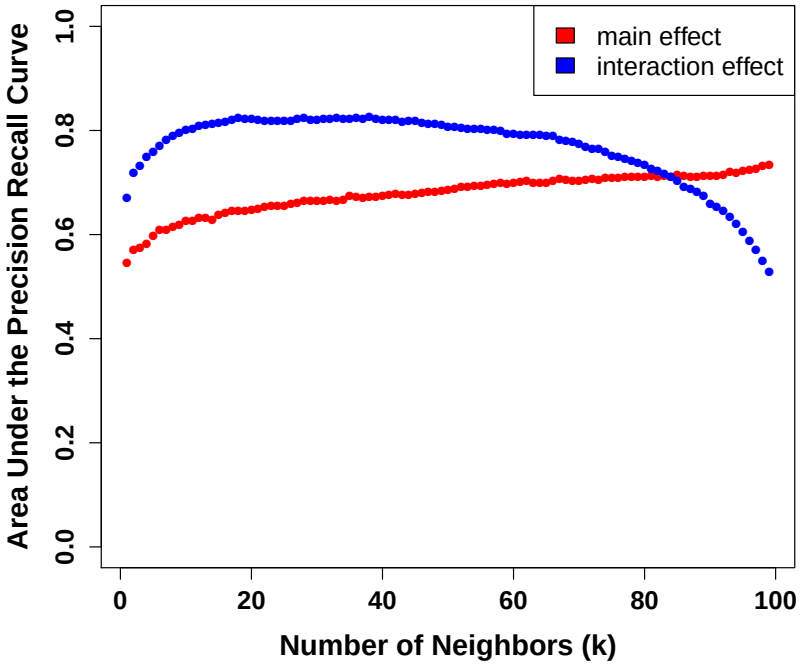

Fig 6. Area Under the Precision-Recall Curve (auPRC) as a function of k for simulated data. Each point on the plot is the average auPRC from 30 simulated data sets with interactions or main effects. Each data set has $m = p = 100$ instances and attributes. Attribute scores were computed with NPDR [7]. The auPRC monotonically increases with increasing k and is maximized at $k = m - 1 = 99$ for main effects. For interactions, auPRC increases between $k = 1$ and $k = 18$ and remains relatively constant until $k = 38$. After $k = 38$, auPRC decreases monotonically between $k = 39$ and $k = 99$.

5 Discussion

References

1. Ryan J. Urbanowicz, Randal S. Olson, Peter Schmitt, Melissa Meeker, and Jason H. Moore. Benchmarking relief-based feature selection methods for bioinformatics data mining. *Journal of Biomedical Informatics*, 85:168–188, 2018.

2. Brett A McKinney, Bill C White, Diane E Grill, Peter W Li, Richard B Kennedy, Gregory A Poland, and Ann L Oberg. ReliefSeq: a gene-wise adaptive-K nearest-neighbor feature selection tool for finding gene-gene interactions and main effects in mRNA-Seq gene expression data. *PloS one*, 8(12):e81527, 2013.

3. Trang T Le, Ryan J Urbanowicz, Jason H Moore, and Brett A McKinney. Statistical inference relief (stir) feature selection. *Bioinformatics*, page bty788, 2018.

4. Caleb A Lareau, Bill C White, Ann L Oberg, and Brett A McKinney. Differential co-expression network centrality and machine learning feature selection for identifying susceptibility hubs in networks with scale-free structure. *BioData mining*, 8(1):5, 2015.

5. Jeffrey T. Leek and John D Storey. Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. *PLoS Genet*, 3(9), September 2007.

6. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

7. Trang T. Le, Bryan A. Dawkins, and Brett A. McKinney. Nearest-neighbor Projected-Distance Regression (NPDR) detects network interactions and controls for confounding and multiple testing. *Under Review*, 2019.