Attribute-wise optimized k with nearest-neighbor projected-distance regression (AWOK-NPDR) for detecting interaction and main effects in high dimensional data: Blessings of Dimensionality and Adaptablity

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 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 and main effects. 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 (Figure 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 (Figure 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.
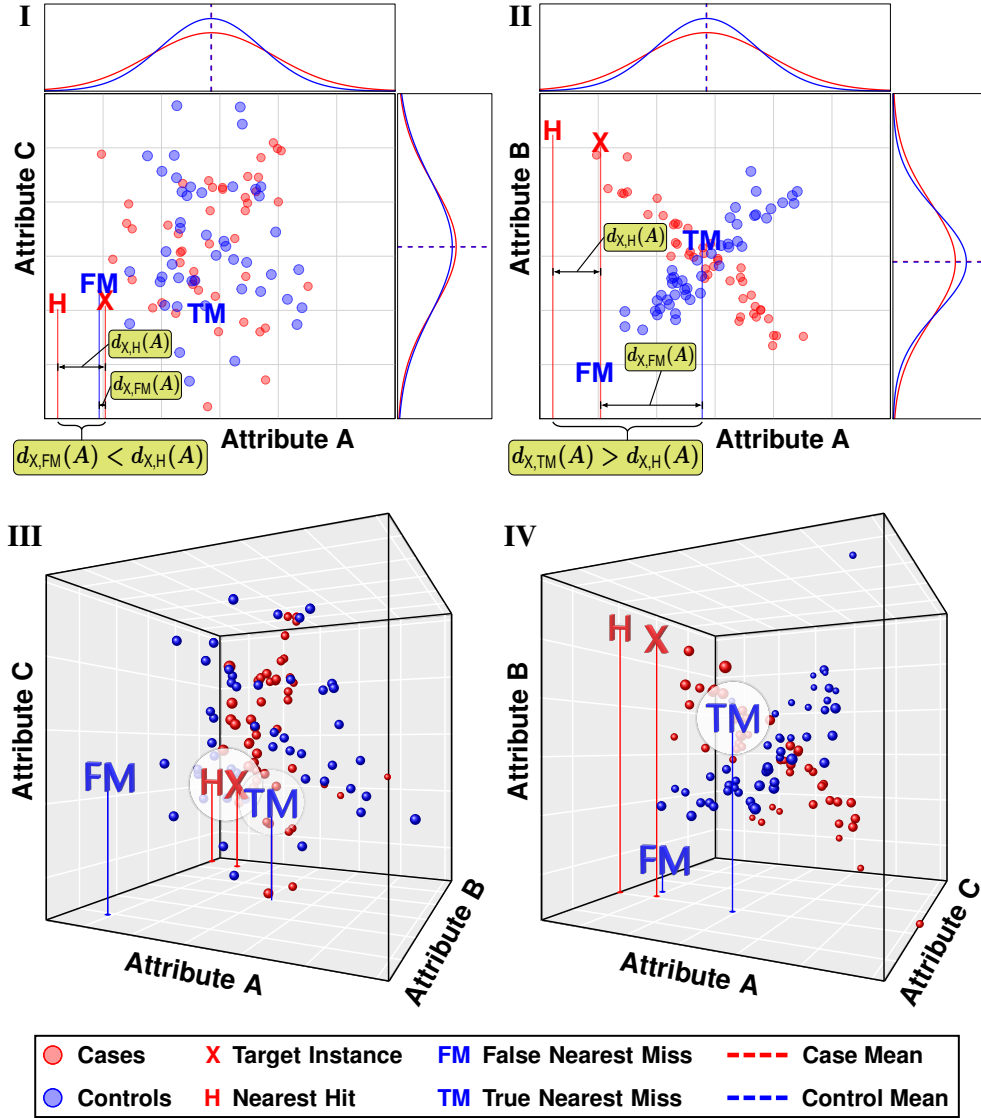
Fig 1. Imposters vs true neighbors in the presence of interactions with three variables. Attributes A, B, and C have no main effect. (**I**) Scatter plot of simulated irrelevant Attribute C with a functional Attribute A. (**II**) Scatter plot of Attributes A and B, which interact through differential 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 Figure 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 (Figure 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 (Figure 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 (Figure 1-III - IV), imposters can be excluded and Relief-based methods will find that A (and B) are important discriminators.
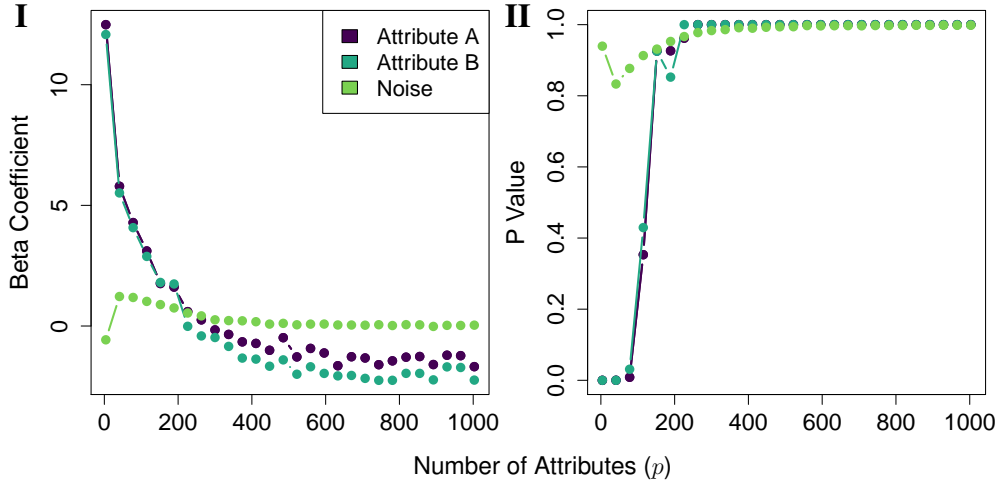

Fig 2. Interacting features A and B in the presence of noise. (**I**) Standardized beta coefficients of interacting features A and B and irrelevant features plotted vs the total number of features p. As more irrelevant features are added to the simulated interacting features A and B, beta coefficients 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 (Figure 2). We iteratively added random null standard normal features and computed standardized beta coefficients (Figure 2-I) and corresponding pseudo P values (Figure 2-II) using a new method called Nearest-neighbor Projected Distance Regression (NPDR) [1]. For binary response variables, like case-control outcomes, NPDR computes a standardized beta coefficient for a generalized linear model (GLM) (Figure 3). The argument to the logit function is $\mathrm{p}_{ij}^{\mathrm{miss}}$, which is the probability that instances i, j are in different phenotype classes. This argument models

the binary outcome diff, which is given by the following

$$d_{ij}^{\mathrm{miss}}(\vec{y}) = \begin{cases} 0 & y_i = y_j, \\ 1 & \text{else.} \end{cases} \tag{1}$$

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 (Figure 2-I) and the average of the corresponding pseudo P values (Figure 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.

<div style="border:1px solid black; padding:10px;">

Nearest-neighbor Projected Distance Regression (NPDR)

For each $a \in \mathcal{A}$ **do**

$$\mathrm{logit}\left(\mathrm{p}_{ij}^{\mathrm{miss}}\right) = \beta_0 + \beta_a \mathrm{d}_{ij}(a) + \epsilon_{ij}, \quad \forall (i,j) \in \mathcal{N}(k)$$

Hypotheses

$$\left. \begin{array}{l} \mathrm{H}_0 : \beta_a \leq 0 \\ \mathrm{H}_1 : \beta_a > 0 \end{array} \right\} \; p_{\mathrm{adj}} < 0.05 \Rightarrow \text{reject } \mathrm{H}_0$$

</div>

Fig 3. Nearest-neighbor Projected Distance Regression for binary response (case-control) [1]. For each attribute $a \in \mathcal{A}$, where $\mathcal{A}$ is the collection of all attributes, we compute standardized beta coefficients for a generalized linear model, where the predictors $(\mathrm{d}_{ij}(a) = |X_{ia} - X_{ja}|)$ are one-dimensional projected distances (diffs) with respect to a particular attribute (a). The argument $\mathrm{p}_{ij}^{\mathrm{miss}}$ is the probability that instances (i, j) are in different phenotype classes. The logit function models the binary hit-miss phenotype projection (Eq. 1). Significant adjusted pseudo P values (p_{adj}) lead to the rejection of the null hypothesis $\beta_a \leq 0$. The set $\mathcal{N}(k)$ is the collection of all neighbor ordered pairs such that each target instances neighborhood has exactly k nearest neighbors.

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 [2], or a attribute-wise optimized k [3]. 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 can change 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 it was shown empirically for balanced case-control datasets that a good constant k approximation to the expected number of neighbors within the multiSURF radii is $k = m/6$ [4], 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

$$D_{ij}^{(q)} = \left(\sum_{a \in \mathcal{A}} |\mathrm{d}_{ij}(a)|^q \right)^{1/q}, \tag{2}$$

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

$$\begin{aligned} \mathrm{d}_{ij}^{\mathrm{num}}(a) &= \mathrm{diff}(a, (i, j)) \\ &= |\hat{X}_{ia} - \hat{X}_{ja}|, \end{aligned} \tag{3}$$

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. NPDR omits the division by $\max(a) - \min(a)$ used by Relief to constrain scores to the interval from -1 to 1 because 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.

The NPDR neighborhood set $\mathcal{N}$ of ordered pair indices is defined 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:

$$\mathcal{N} = \{\{(i, j)\}_{i=1}^m\}_{\{j \neq i : j \in N_i\}}. \tag{4}$$

The cardinality of the set $\{j \neq i : j \in N_i\}$ is k_i, the number of nearest neighbors for subject i. For fixed k, this cardinality is just $k_i = k$ for each target instance $i \in \mathcal{I}$.

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

$$D_{ij} \leq R_i^\alpha \implies j \in N_i^\alpha, \tag{5}$$

where the threshold radius for instance i is

$$R_i^\alpha = \bar{D}_i - \alpha\,\sigma_{\bar{D}_i} \tag{6}$$

and

$$\bar{D}_i = \frac{1}{m-1}\sum_{j\neq i} D_{ij}^{(\cdot)} \tag{7}$$

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

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 number of neighbors is given by

$$\bar{k}_\alpha = (m-1)q_\alpha, \tag{8}$$

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 Figure 4. This integral can be written in terms of the error function (erf):

114
115
116
117
118
119
120
121
122
123

$$q_\alpha = \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{\alpha}{\sqrt{2}}\right)\right). \tag{9}$$

And finally using Eqs. (8 and 9) we find

$$\bar{k}_\alpha = \left\lfloor \frac{m-1}{2}\left(1 - \operatorname{erf}\left(\frac{\alpha}{\sqrt{2}}\right)\right)\right\rfloor, \tag{10}$$

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 $\bar{k}_\alpha^{\text{hit/miss}} = \bar{k}_{1/2}^{\text{hit/miss}} = \frac{1}{2}\bar{k}_{1/2} = .154(m-1)$, 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.
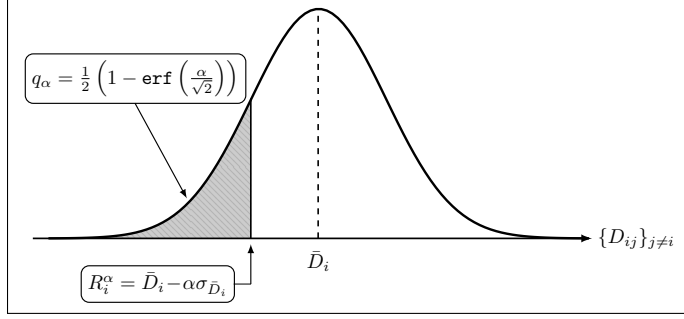
125
126
127
128
129
130
131
132

Fig 4. 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 [5]. 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 5). Analogous to the former method [5], we start with a random graph with either Erdös-Rényi or Scale-free degree distribution (Figure 5**(1)**). From the random graph, we choose which attributes will be given functional interactions (F) among those with non-zero degree (Figure 5**(2)**). Using adjacency matrix A, along with the indices of interaction attributes F, we create correlation matrices for cases and controls (Figure 5**(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

$$b^{\text{int}} = -t\rho^{\text{hi}} + (1-t)\rho^{\text{hi}}, \text{ where } t \in [0,1]. \tag{11}$$

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 [5].

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 5(**4**)). We then simulate null data matrices X^{case} and X^{ctrl} (Figure 5(**5**)) for cases and controls, respectively, such that

$$x_{ij}^{\text{case}}, x_{ij}^{\text{ctrl}} \sim \mathcal{N}(0,1) \quad \forall i,j. \tag{12}$$

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 $m \times p$ data matrix given by

$$X^{\text{int}} = \left[\begin{array}{c} Y^{\text{ctrl}} \\ \hline Y^{\text{case}} \end{array} \right]. \tag{13}$$

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 5(**1**)(**5**)) to generate only interactions, or we can use the final step of the algorithm (Figure 5(**5**)) in combination with main effect simulations to create data with mixed effects (Figure 5(**6**)-(**7**)). We explain the main effect simulations (Figure 5(**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 [6]. This method uses a linear model given by

$$X_{ia} = b_a y_i + \varepsilon_{ia}, \tag{14}$$

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 5(**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 5(**5**)) and main effect simulations (Figure 5(**6**)) into a single data set (X) (Figure 5(**7**)), where the left block matrix of X contains interactions and noise and the right block of X contains only main effects. Our mixed effects (interactions + main effects) simulation method can be used to create a wide variety of different types of data sets. In particular, our interaction simulation strategy is novel in that we simulate correlation matrices for cases and controls directly. We tune the interaction effect size by adjusting ρ^{hi}, b^{int} (Eq. 11), and the degree of connectivity of interacting attributes in the random graph (Figure 5(**1**)).
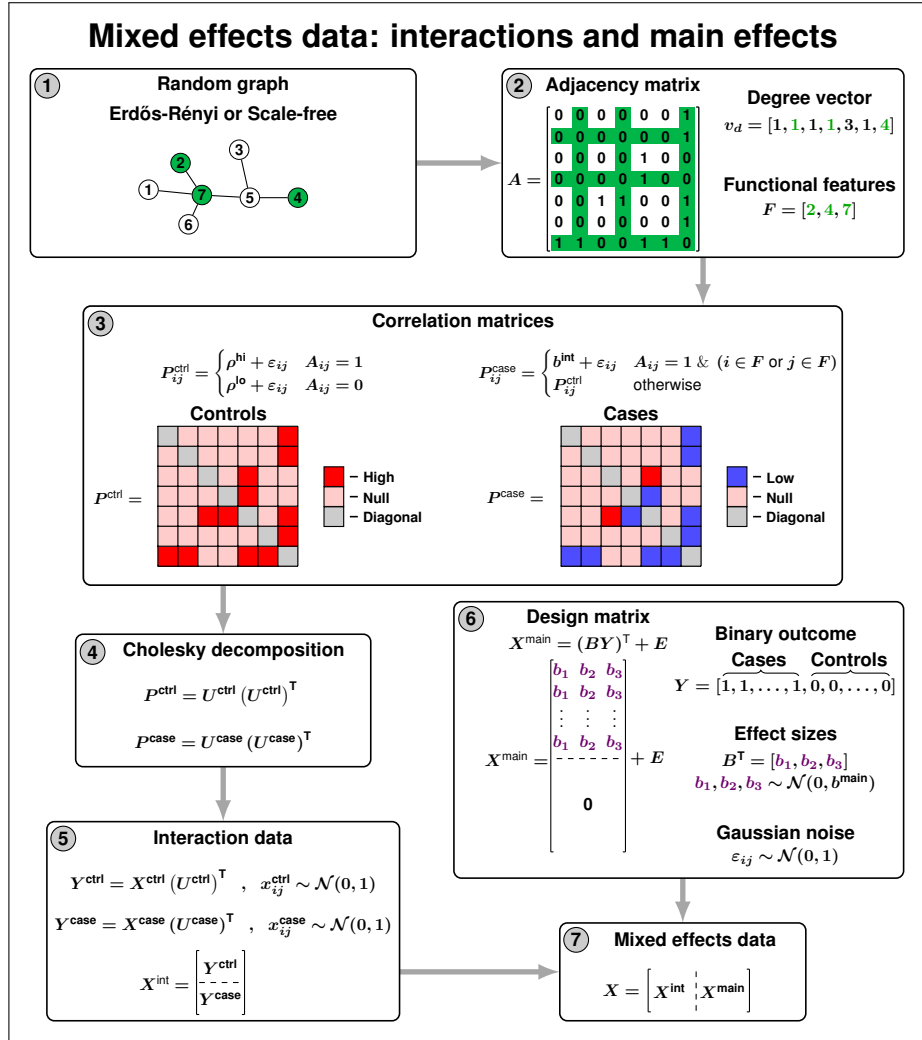
Fig 5. Algorithm for simulating mixed effects data with interactions and main effects. **(1)** Random network is generated, whose degree distribution is either Erdős-Rényi or Scale-free. **(2)** Adjacency matrix (A) and degree vector (v_d) corresponding to the random network are computed and functional 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 $(X^{\text{ctrl}}$ and $X^{\text{case}})$ 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^{int}) and main effects (X^{main}) combined X.

4 Optimizing k for detecting effects

We use a method that we call Attribute-Wise Optimized k (AWOK), formerly referred to as Gene-wise Adaptive k (GWAK) [3], to determine the optimal k for each attribute

in a particular simulated data set (Figure 6). AWOK 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 AWOK method can be applied similarly using any nearest-neighbor feature selection algorithm, such as Relief-Based Algorithms [2]. The success of AWOK 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 AWOK is because we can explore all values of k, from $k = 1$ to $k = m - 1$, for each attribute.

Fig 6. Attribute-Wise Optimized k (AWOK) 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.

It is important for feature selection methods to simultaneously detect as many outcome-relevant attributes as possible and also minimize the number of false positives that have no causal relationship with the outcome. To maximize relevant attribute detection for a given feature selection algorithm, we must maximize its recall with respect to attribute scores (Eq. 15). Recall is maximized by selecting as many true positives (TP) as possible, leaving as few false negatives (FN) as possible. Precision (Eq. 16) is maximized by selecting as many true positives (TP) as possible, while minimizing the number of false positives (FP) among selected attributes.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{15}$$

where TP is the total number of attributes correctly labeled positive (relevant to outcome) and FN is the total number of attributes incorrectly labeled negative (irrelevant to outcome).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{16}$$

where TP is the total number of attributes correctly labeled positive (relevant to outcome) and FP is the total number of attributes incorrectly labeled positive. We show an example of how precision and recall are computed for an artificial data set (Fig. 7). Attribute scores are sorted from largest to smallest, where scores are standardized NPDR beta coefficients in this case. Each attribute has a label that is either 1 (functional) or 0 (noise). As larger percentages of the top scored attributes are selected, precision tends to decrease because we select more false positives (FP). The opposite is true for recall, where there is increasing behavior by forcing the number of false negatives (FN) to zero. After all values of precision and recall are computed for each selection percentage of the ranked attributes, we plot the precision-recall curve (PRC). The area under the precision-recall curve (auPRC) indicates how well the method detects relevant attributes (optimizes recall) and controls for false positives (optimizes precision).

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.
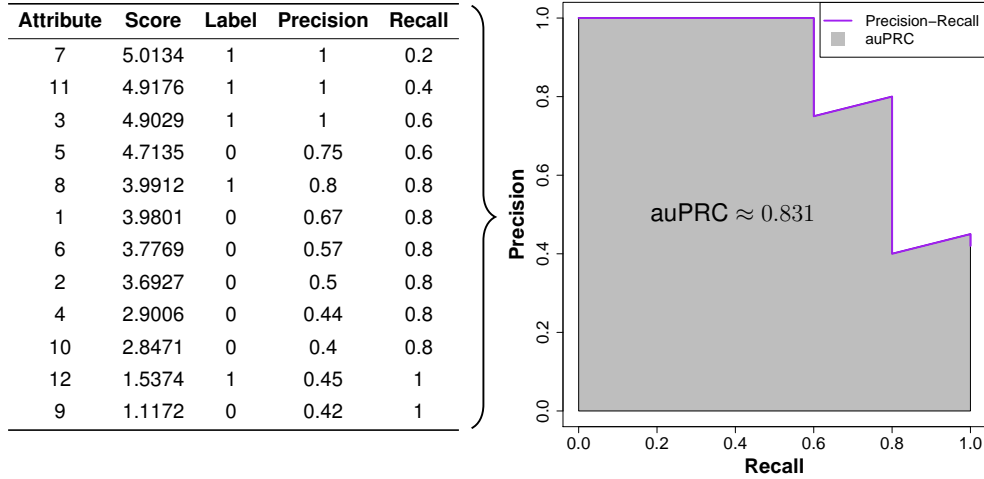
Attribute	Score	Label	Precision	Recall
7	5.0134	1	1	0.2
11	4.9176	1	1	0.4
3	4.9029	1	1	0.6
5	4.7135	0	0.75	0.6
8	3.9912	1	0.8	0.8
1	3.9801	0	0.67	0.8
6	3.7769	0	0.57	0.8
2	3.6927	0	0.5	0.8
4	2.9006	0	0.44	0.8
10	2.8471	0	0.4	0.8
12	1.5374	1	0.45	1
9	1.1172	0	0.42	1

Fig 7. Example of how we compute precision and recall from attribute scores in simulated data. After applying NPDR, we sort attribute scores from largest to smallest. The binary label for each attribute represents functionality. If the label is 1, then the attribute is causally related to the outcome. If the label is 0, then the attribute is just noise. After computing precision and recall, we plot the precision-recall curve (PRC). The area under this curve is known as area under the Precision-Recall Curve (auPRC).

For each value of k, we plotted the average NPDR beta and auPRC from the 30 simulation replicates for both main effects and interaction effects (Figure 8). We also plotted the average beta coefficient for noise attributes (Figure 8**A**). Main effect beta coefficients increase monotonically with increasing k (Figure 8**A**). We also observed that auPRC for data sets with main effects increases monotonically with increasing k (Figure 8**B**) Interaction effect beta coefficients are maximized within an intermediate range of k, where there is increasing behavior between $k = 1$ and $k = 18$ and decreasing behavior between $k = 39$ and $k = 99$ (Figure 8**A**). We see the same behavior in auPRC for interactions (Figure 8**B**). The fact that NPDR beta coefficients for interactions and main effects have the same behavior as auPRC as a function of k indicates that optimizing NPDR beta coefficients for each individual attribute is a legitimate strategy

to optimize feature quality. This is additionally supported by the fact that noise attribute NPDR beta coefficients remain approximately 0 for all k. Such different detectability between interactions and main effects 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 AWOK, we can tune k for each individual attribute to avoid having to choose between the different types of effects.
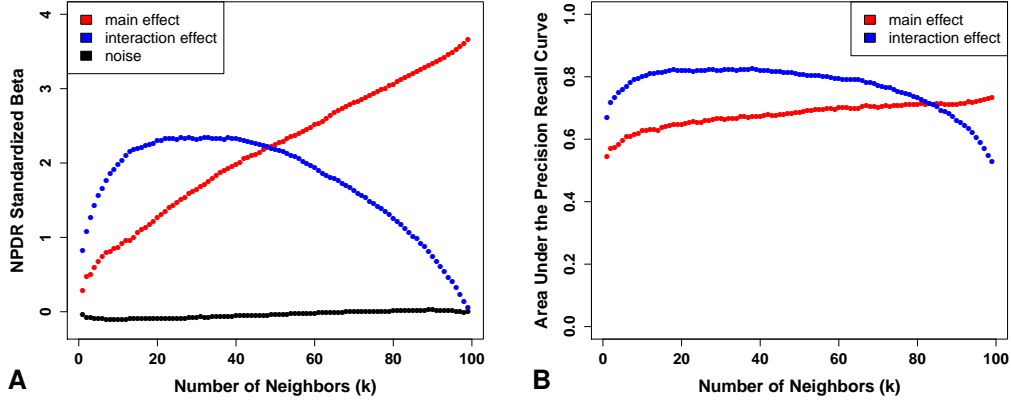

Fig 8. (**A**) NPDR standardized beta coefficient as a function of k [1]. Each point on the plot is the average NPDR beta coefficient from 30 simulated data sets with interactions or main effects. Each data set has $m = p = 100$ instances and attributes. For interactions, the beta coefficient increases between $k = 1$ and $k = 18$ and remains relatively constant until $k = 38$. After $k = 38$, the beta coefficient decreases monotonically between $k = 39$ and $k = 99$. For main effects, the beta coefficient increases monotonically with increasing k. For noise, the beta coefficient does not change significantly with varying k. (**B**) Area Under the Precision-Recall Curve (auPRC) as a function of k for the same simulated data. The behavior of auPRC and the NPDR beta coefficients as a function of k are similar.

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 AWOK without splitting the data into training and test folds. Generally, one would intuitively apply a cross-validation method, such as nested cross-validation (nested-CV), for parameter tuning to avoid overfitting (Figure 9).

However, we have seen empirically that cross-validation does not produce high quality attributes in the context of nearest-neighbor feature selection. We compared the Area Under the Precision-Recall Curve (auPRC) for Global best fixed k, AWOK, and (nested-CV) on simulated data (Figure 10). All simulation scripts for interactions, main effects, and mixed effects, were written in **R** [7], and are available on Github at the following link (https://github.com/insilico/npdr). Using our simulation method (Figure 5), we generated 30 data sets with main effects and 30 data sets with interaction effects. The best auPRC was computed for each data set with respect to k using each of these methods. For a given data set, Global best fixed k simply loops through each possible value of k, computes attribute scores with a nearest-neighbor feature selection algorithm, and chooses the value of k that maximizes auPRC.

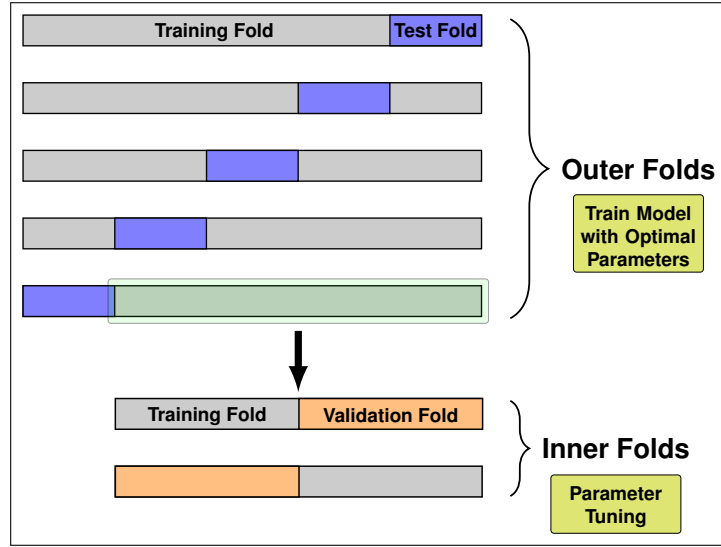One reason that cross-validation to tune k does not perform well in nearest-neighbor

Fig 9. Nested cross-validation algorithm for simultaneously tuning parameters and selecting models. The data is split into training and test folds, where each outer training fold is further split into training and validation folds in order to tune parameters and select features. Parameters and features are chosen in the inner folds to optimize validation accuracy. Selected features and tuned parameters from the inner loop are used to train on the full outer training fold. Features and tuned parameters corresponding with the highest outer test accuracy are ultimately selected.

feature selection algorithms is because it optimizes classification accuracy and does not optimize attribute scores. Attribute scores generated from nearest-neighbor feature selection algorithms are fundamental to their success. Furthermore, feature quality is sensitive to changes in neighbors. Cross-validation methods partition data into training and test sets, which limits the possible neighbors for any given target instance within a training fold. For all of these reasons, the output from the nested-CV algorithm for tuning k is a collection of mostly irrelevant attributes, which implies that the method is overfitting the training data.

Using the same 30 simulated data sets with either interactions or main effects, we compared auPRC between AWOK, MultiSURF (MSURF), and MultiSURF average k ($\bar{k}_\alpha$, Eq. 10) (Figure 11). As in the comparison of AWOK with nested-CV and Global best k, we see that AWOK outperforms the other methods by consistently giving a higher auPRC on simulated data. The imbalance in interaction and main effect detection that is seen with MultiSURF vanishes with AWOK. The performances of adaptive radius MultiSURF and MultiSURF average k ($\bar{k}_\alpha$, Eq. 10) are essentially the same. This provides further justification for using AWOK in nearest-neighbor feature selection algorithms. Equivalence in auPRC between MultiSURF and our average k is an important validation, indicating that our formula (Eq. 10) works in the context of data with interactions and main effects. It is important to note from the comparison of nested-CV and AWOK (Figure 10) and the comparison of MultiSURF and AWOK (Figure 11) that AWOK is the most adaptive of the three different methods. This is due to the fact that each attribute has its own value of k in AWOK. MultiSURF is slightly less adaptive since it only allows each target instance a different radius. In the case of nested-CV, only a single value of k is chosen for a given data set so there is no adaptability. Because the performance of AWOK is better than all of the comparison methods with respect to auPRC and this metric monotonically increases with increased neighborhood adaptability, the optimal strategy for choosing k seems to be that which
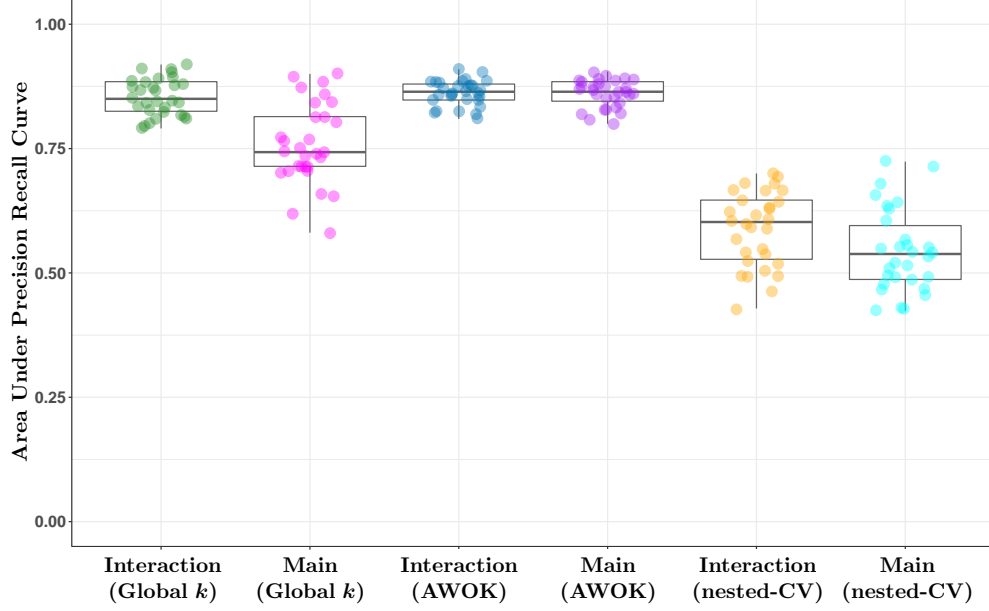
Fig 10. Comparison of Area Under the Precision-Recall Curve (auPRC) for three methods on simulated data with main effects or interactions. Each point on any particular boxplot represents the max auPRC for one of 30 simulated data sets, where auPRC is maximized as a function of k. We compared auPRC for each data set with respect to Global fixed k, Attribute-Wise Optimized k (AWOK), and nested cross-validation (nested-CV). AWOK has the best performance overall and the lowest variance in auPRC. Unlike the other methods, AWOK auPRC is balanced between interactions and main effects because each attribute has its own best k.

maximizes adaptability.

To explore how the optimal choice of k changes in different contexts, we generated simulation replicates for several different scenarios on the data. Each simulated data set $X^{(m \times p)}$ generated contained $m = 100$ instances and $p = 1000$ features. 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. We also considered different ratios of main-to-interaction effect among functional attributes. Ratios of 0.2, 0.4, 0.6, and 0.8 of main-to-interaction, where 0.2 means that 20% of the functional attributes are main effects while 80% are interactions, were considered. For each simulation, we kept the main effect size parameter (b^{main}, Figure 5(**6**)) 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}, Figure 5(**3**)) only. The effect size increases monotonically with increasing ρ^{hi} because b^{int} is simply a point on the line from $-\rho^{\mathrm{hi}}$ to ρ^{hi}, which is controlled by an input t in our simulation **R** function. We considered values of $\rho^{\mathrm{hi}} \in \{0.3, 0.6, 0.9\}$ so that we can explore small, medium, and large interaction effect sizes across all scenarios. The non-connection, low correlation parameter (ρ^{lo}, Figure 5(**3**)) was fixed at 0.1 for all simulations. Taking all of the simulation parameters into account, we have a total of 12 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 ran AWOK on each simulated data set to compute a value of k for each attribute.
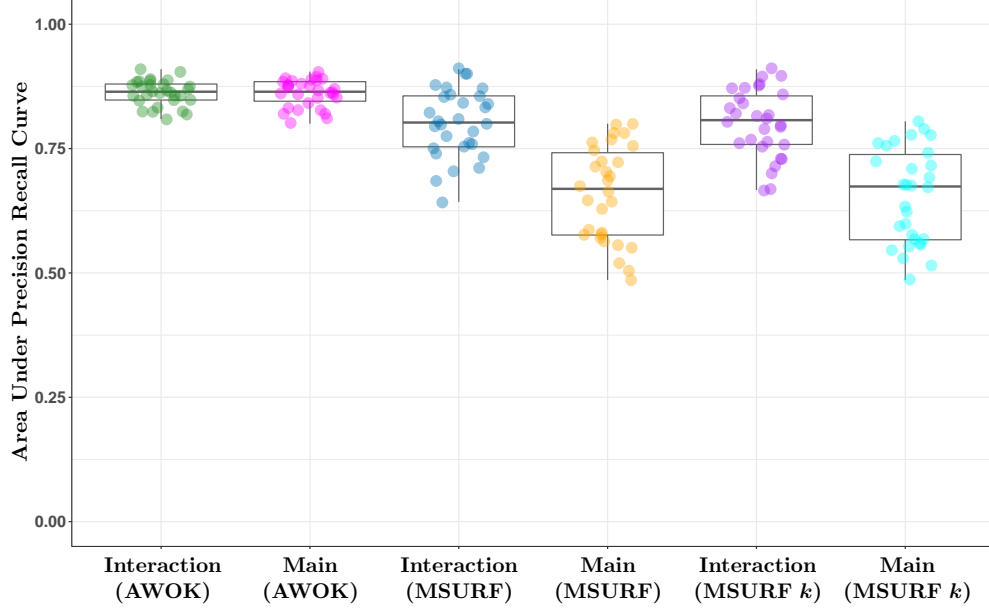
Fig 11. Comparison of Area Under the Precision-Recall Curve (auPRC) for three methods on simulated data with main effects or interactions. Each point on any particular boxplot represents the max auPRC for one of 30 simulated data sets, where auPRC is maximized as a function of k. We compared auPRC for each data set with respect to Attribute-Wise Optimized k (AWOK), adaptive radius method MultiSURF (MSURF), and predicted MultiSURF fixed k (MSURF k). AWOK has the best performance overall and the lowest variance in auPRC. Unlike the other methods, AWOK auPRC is balanced between interactions and main effects because each attribute has its own best k.

Across all iterations for a given simulation scenario, the average AWOK value of k that maximized the scores of interactions ($k_{\text{ave}}^{\text{int}}$), main effects ($k_{\text{ave}}^{\text{main}}$), and noise attributes ($k_{\text{ave}}^{\text{noise}}$) was computed. We also computed the average Global best k ($k_{\text{ave}}^{\text{auPRC}}$), which is the average value of k across all iterations of a given simulation scenario that maximized auPRC. For reasons we have already mentioned, we consider the AWOK values of k to be the best choice for a given data set. Therefore, the average AWOK values of k we computed are considered optimal for each simulation scenario. We give a summary from all of our simulation scenarios that shows the different values of k for AWOK, Global best k, and MultiSURF (Table 1).

5 Discussion

In our analysis of optimizing the neighborhood selection parameter k in nearest-neighbor feature selection algorithms, we have determined that there is no single value of k that produces the best set of attributes from a given data set. The value of k should adapt to each attribute in the data in order to maximize the ability to detect those that are relevant to a given outcome. Adaptability in neighborhood selection parameters, whether it is attribute or instance adaptability, seems to be the defining characteristic of successful methods. For adaptive radius methods like MultiSURF [2], each target instance has its own neighborhood with a radius that is derived from the average pairwise distance to all other instances $j \neq i$. This method performs well in detecting interactions, so we derived

Table 1. Summary of average fixed k for three different methods. Using AWOK (Figure 6), we computed the average optimized k for interactions ($k_{\text{ave}}^{\text{int}}$), main effects ($k_{\text{ave}}^{\text{int}}$), and noise attributes ($k_{\text{ave}}^{\text{noise}}$). We also applied Global best k to optimize auPRC ($k_{\text{ave}}^{\text{auPRC}}$). The average MultiSURF k ($k_{\text{ave}}^{\text{msurf}}$) is given for comparison. The high correlation parameter (ρ^{hi}, Fiure 5(**3**)) was varied within $\{0.3, 0.6, 0.9\}$ to explore low, medium, and high interaction effect sizes. Mix Ratio represents the ratio of main-to-interaction effects among functional attributes, where 0.2 means 20% main effects and 80% interactions.

$\rho^{\mathbf{hi}}$	**Mix Ratio**	$k_{\text{ave}}^{\text{int}}$	$k_{\text{ave}}^{\text{main}}$	$k_{\text{ave}}^{\text{noise}}$	$k_{\text{ave}}^{\text{auPRC}}$	$k_{\text{ave}}^{\text{msurf}}$
0.3	0.2	28	61	31	27	32
0.3	0.4	28	61	31	27	32
0.3	0.6	28	61	31	27	32
0.3	0.8	28	61	31	27	32
0.6	0.2	29	58	31	27	32
0.6	0.4	29	58	31	27	32
0.6	0.6	29	58	31	27	32
0.6	0.8	29	58	31	27	32
0.9	0.2	28	61	31	27	32
0.9	0.4	28	61	31	27	32
0.9	0.6	28	61	31	27	32
0.9	0.8	28	61	31	27	32

the corresponding fixed k that approximates the average MultiSURF neighborhood order ($\bar{k}_{1/2}$). The distribution of pairwise distances between pairs of instances $i, j \in \mathcal{I}$ is approximately normal for even relatively small numbers of attributes (p). The normal approximation holds when pairwise correlation between attributes is relatively low, which is true in many real data sets. Because the number of instances within a fixed radius about a target instance is a random Binomial variable with $n = 2$ trials and some neighborhood inclusion probability of "success" (q), we were able to derive the average number of neighbors within a given fixed radius neighborhood, where q is derived from the normal approximation to a given distance distribution. MultiSURF uses a neighborhood radius that is a function of the average target-to-neighbor distance and the standard deviation of the target-to-neighbor distances. Specifically, the MultiSURF radius for a given target instance is the average target-to-neighbor distance minus one-half the standard deviation of the target-to-neighbor distances ($\alpha = 1/2$,Eq. 6). We generalized our approximation of neighborhood order to any α radius about target instances for adaptive radius algorithms that compute target-to-hit and target-to-miss neighborhoods either separately (Eq. 10) or without distinction (Eq. 8). These estimates hold if the distance distribution is approximately normal, which is true in the case of many real data sets.

We developed a novel strategy for simulating interaction effects (Figure 5(**1**)-(**5**)) in order to have fine-tuned control over interaction effect sizes. Our simulation method starts from a random graph that has either Erdös-Rényi or Scale-free degree distribution (Figure 5(**1**)). We choose interacting attributes (F) among those with positive degree (v_d) in the graph (Figure 5(**2**)). The most novel aspect of this simulation method is that we generate case and control pairwise attribute correlation matrices directly (Figure 5(**3**)). Differential correlation is creating by adjusting the pairwise correlations between attributes chosen for interaction effect (F) and their connections from the random graph. We have the ability to choose the level of connectivity for each attribute in the random graph, as well as the strength of connections (ρ^{hi}). The magnitude

of the differential correlation for interactions (b^{int}, Eq. 11) is a function of the high correlation parameter (ρ^{hi}). Specifically, b^{int} is a point on the line from $-\rho^{\text{hi}}$ to ρ^{hi}. Cumulatively, this allows us to control interaction effect sizes by adjusting network connectivity, connection strength (ρ^{hi}), and magnitude differential correlation (b^{int}, Eq. 11). To create data with a mixture of interactions and main effects, we combine our interaction simulation strategy with a main effect simulation method that was previously developed for gene expression data [6]. This main effect simulation method allows us to control effect sizes as well (b^{main}, Figure 5**(6)**). We combine interaction effects and main effects as left and right block matrices, respectively (Figure 5**(7)**).

Using our simulation method, we determined empirically that a method called Variable - Wise Optimized k (AWOK, Figure 6) outperforms other methods like MultiSURF, nested cross-validation (nested-CV, Figure 9), Global best fixed k, and predicted MultiSURF average k ($\alpha = 1/2$, Eq. 10) with respect to maximizing the Area Under the Precision-Recall Curve (Figures 10-11). The standard procedure to avoid overfitting is to apply cross-validation methods. However, we showed that features selected through nested-CV were low in quality as compared to all other methods. This is largely due to the fact that cross-validation methods optimize test set classification accuracy from a given model instead of optimizing attribute scores. Based on our analysis, it may be problematic to use cross-validation methods when a nearest-neighbor feature selection algorithm is being used. We did not consider the problem of model selection in this work, which is commonly an additional goal along with feature selection. Our main focus was on tuning k to maximize the probability of detecting functional attributes among those with the highest scores. Because of this goal, we determined that AWOK is a reliable strategy for scoring attributes with nearest-neighbor feature selection algorithms. Other methods seem to be slightly better in detecting interactions than main effects, although this could just be our simulation parameterization, with larger variance in auPRC. We showed with simulations that main effects require larger values of k, whereas interactions require more intermediate values (Figure 8). AWOK tunes k for each attribute so that one does not need to accept the trade off of detecting main effects at the expense of interactions (or vice versa).

We constructed a variety of simulation scenarios to summarize the optimal k for each, as determined by AWOK (Table 1). Varying the high correlation parameter of our simulation method (ρ^{hi}), as well as the mixture of interactions and main effects among functional attributes, we were able to cover several different combinations of effect sizes for interactions and main effects. The average value of k that optimized detection of interactions ($k_{\text{ave}}^{\text{int}}$) and main effects ($k_{\text{ave}}^{\text{main}}$) was consistent with our conclusion that interactions require intermediate values of k and main effects require larger values of k (Figure 8). The average AWOK k for noise attributes did not change from one scenario to the next. The fixed value of k that optimized auPRC across all simulated data sets ($k_{\text{ave}}^{\text{auPRC}}$) seemed to favor the detection of interactions because we saw that $k_{\text{ave}}^{\text{auPRC}} \approx k_{\text{ave}}^{\text{int}}$ in each simulation scenario. The MultiSURF average k ($k_{\text{ave}}^{\text{msurf}}$) also seemed to be tuned more towards detecting interactions, but was slightly inflated due to the pairwise correlation between attributes determined by the high correlation parameter (ρ^{hi}).

The fundamental conclusion of our analysis in optimizing k is that there is no single value of k that should necessarily be considered a rule of thumb in nearest-neighbor feature selection. A similar conclusion is warranted for radius methods as well. The neighborhood order should be either adaptive to each attribute, as in AWOK, or adaptive to each instance (e.g., MultiSURF). The performance of AWOK as compared to other methods seems to indicate, at least empirically in simulated data, that adaptive k methods are among the best in detecting relevant attributes in high dimensional data.

References

1. Trang T Le, Bryan A Dawkins, and Brett A McKinney. Nearest-neighbor Projected-Distance Regression (NPDR) for detecting network interactions with adjustments for multiple tests and confounding. *Bioinformatics*, 01 2020. btaa024.

2. 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.

3. 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.

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

5. 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.

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

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