Noname manuscript No.
(will be inserted by the editor)

Probabilistic Characterization of Nearest Neighbor Classifier

Amit Dhurandhar · Alin Dobra

Received: date / Accepted: date

Abstract The k-Nearest Neighbor classification algorithm (kNN) is one of the most simple yet effective classification algorithms in use. It finds major applications in text categorization, outlier detection, handwritten character recognition, fraud detection and in other related areas. Though sound theoretical results exist regarding convergence of the Generalization Error (GE) of this algorithm to Bayes error, these results are asymptotic in nature. The understanding of the behavior of the kNN algorithm in real world scenarios is limited. In this paper, assuming categorical attributes, we provide a principled way of studying the non-asymptotic behavior of the kNN algorithm. In particular, we derive exact closed form expressions for the moments of the GE for this algorithm. The expressions are functions of the sample, and hence can be computed given any joint probability distribution defined over the input-output space. These expressions can be used as a tool that aids in unveiling the statistical behavior of the algorithm in settings of interest viz. an acceptable value of k for a given sample size and distribution. Moreover, Monte Carlo approximations of such closed form expressions have been shown in [6,5] to be a superior alternative in terms of speed and accuracy when compared with computing the moments directly using Monte Carlo. This work employs the semi-analytical methodology that was proposed recently to better understand the non-asymptotic behavior of learning algorithms.

Keywords kNN · moments

1 Introduction

A major portion of the work in Data Mining caters towards building new and improved classification algorithms. Empirical studies [10,14,8,17,9,20] and theoretical results, mainly asymptotic [24,19,2] in nature, assist in realizing this endeavor. Though both

Amit Dhurandhar
IBM TJ Watson
E-mail: adhuran@us.ibm.com

Alin Dobra
University of Florida
E-mail: adobra@cise.ufl.edu

methods are powerful in their own right, results from empirical studies heavily depend on the available datasets and results from the theoretical studies depend on a large and mostly unknown dataset size after which the asymptotic results reasonably apply. Keeping these factors in mind, a semi-analytical methodology was proposed in [6] to study the non-asymptotic behavior of classification algorithms. Previously, the methodology was applied to characterize the naive bayes classifier and random decision trees. It was suggested that applying such a methodology to characterize other commonly used algortihms would be a challenging, but useful endeavour. With this in mind, we in this paper, characterize the popular k-nearest neighbor (kNN) algorithm for the discrete case.

The kNN algorithm is a simple yet effective and hence commonly used classification algorithm in industry and research [7, 16, 15, 3, 21, 25, 13, 18]. It is used in many applications in domains such as bioinformatics [21], time series analysis [25, 13], geology [18], chemical engineering [15], etc.

The kNN algorithm is known to be a consistent estimator [23], i.e. it asymptotically achieves Bayes error within a constant factor. None of the even more sophisticated classification algorithms eg. SVM, Neural Networks etc. are known to outperform it consistently [22]. However, the algorithm is susceptible to noise and choosing an appropriate value of k is more of an art than science.

In the next few subsections, we briefly describe our methodology. We then enumerate the specific contributions that we make in this paper.

1.1 *What* is the methodology ?

The methodology for studying classification models consists in studying the behavior of the first two central moments of the GE of the classification algorithm studied. The moments are taken over the space of all possible classifiers produced by the classification algorithm, by training it over all possible datasets sampled i.i.d. from some distribution. The first two moments give enough information about the statistical behavior of the classification algorithm to allow interesting observations about the behavior/trends of the classification algorithm with respect to (w.r.t.) any chosen data distribution.

1.2 *Why* have such a methodology?

The answers to the following questions shed light on why the methodology is necessary if tight statistical characterization is to be provided for classification algorithms.

1. *Why study GE ?* The biggest danger of learning is *overfitting* the training data. The main idea in using GE – the expected error over the entire input, as a measure of success of learning, instead of empirical error on a given dataset, is to provide a mechanism to avoid this pitfall.
2. *Why study the moments instead of the distribution of GE ?* Ideally, we would study the distribution of GE instead of moments in order to get a complete picture of what is its behavior. Studying the distribution of discrete random variables, except for very simple cases, turns out to be very hard. The difficulty comes from the fact that even computing the pdf in a single point is intractable since all combinations of random choices that result in the same value for GE have to be enumerated. On the

other hand, the first two central moments coupled with distribution independent bounds such as Chebychev and Chernoff give guarantees about the worst possible behavior that are not too far from the actual behavior (small constant factor). Interestingly, it is possible to compute the moments of a random variable like GE without ever explicitly writing or making use of the formula for the pdf. What makes such an endeavor possible is extensive use of the linearity of expectation as explained in [6].

3. *Why characterize a class of classifiers instead of a single classifier ?* While the use of GE as the success measure is standard practice in Machine Learning, characterizing classes of classifiers instead of the particular classifier produced on a given dataset is not. From the point of view of the analysis, without large testing datasets it is not possible to evaluate directly GE for a particular classifier. By considering classes of classifiers to which a classifier belongs, an indirect characterization is obtained for the particular classifier. This is precisely what Statistical Learning Theory (SLT) does; there the class of classifiers consists in all classifiers with the same VC dimension. The main problem with SLT results is that classes based on VC dimension are too large, thus results tend to be pessimistic. In the methodology in [6], the class of classifiers consists only of the classifiers that are produced by the given classification algorithm from datasets of fixed size from the underlying distribution. This is the probabilistic smallest class in which the particular classifier produced on a given dataset can be placed in.

1.3 *How* do we implement the methodology ?

We estimate the moments of GE, by obtaining parametric expressions for them. If this can be accomplished the moments can be computed exactly. Moreover, by dexterously observing the manner in which expressions are derived for a particular classification algorithm, insights can be gained into analyzing other algorithms. Though deriving the expressions may be a tedious task, using them we obtain highly accurate estimates of the moments. In this paper, we analyze the kNN algorithm applied to categorical data. The key to the analysis is focusing on how the algorithm builds its final inference. In cases where the parametric expressions are computationally intensive to compute exactly, the approximations proposed in Section 5 can be used to obtain estimates with small error.

If the moments are to be studied on synthetic data then the distribution is anyway assumed and the parametric expressions can be directly used. If we have real data an empirical distribution can be built on the dataset and then the parametric expressions can be used.

1.4 Applications of the methodology

It is important to note that the methodology is not aimed towards providing a way of estimating bounds for GE of a classifier on a given dataset. The primary goal is creating an avenue in which learning algorithms can be studied precisely i.e. studying the statistical behavior of a particular algorithm w.r.t. a chosen/built distribution. Below, we discuss the two most important perspectives of applying the methodology.

1.4.1 Algorithmic Perspective

If a researcher/practitioner designs a new classification algorithm, one needs to validate it. Standard practice is to validate the algorithm on a relatively small (5-20) number of datasets and to report the performance. By observing the behavior of only a few instances of the algorithm the designer infers its quality. Moreover, if the algorithm under performs on some datasets, it can be sometimes difficult to pinpoint the precise reason for its failure. If instead one is able to derive parametric expressions for the moments of GE, the test results would be more relevant to the particular classification algorithm, since the moments are over all possible datasets of a particular size drawn independently and identically from some chosen/built distribution. Testing individually on all these datasets is an impossible task. Thus, by computing the moments using the parametric expressions the algorithm would be tested on a plethora of datasets with the results being highly accurate. Moreover, since the testing is done in a controlled environment i.e. all the parameters are known to the designer while testing, one can precisely pinpoint the conditions under which the algorithm performs well and the conditions under which the algorithm under performs.

1.4.2 Dataset Perspective

If an algorithm designer validates his/her algorithm by computing moments as mentioned earlier, it can instill greater confidence in the practitioner searching for an appropriate algorithm for his/her dataset. The reason for this being, if the practitioner has a dataset which has a similar structure or is from a similar source as the test dataset on which an empirical distribution was built and favorable results reported by the designer, then this would mean that the results apply not only to that particular test dataset, but to other similar type of datasets and since the practitioner's dataset belongs to this similar collection, the results would also apply to his. Note that a distribution is just a weighting of different datasets and this perspective is used in the above exposition.

1.5 Specific Contributions

In this paper, we develop expressions for the first 2 moments of GE for the k-Nearest Neighbor classification algorithm built on categorical data. We accomplish this by expressing the moments as functions of the sample produced by the underlying joint distribution. In particular, we develop efficient characterizations for the moments when the distance metric used in the kNN algorithm, is independent of the sample. We also discuss issues related to the scalability of the algorithm. We use the derived expressions, to study the classification algorithm in settings of interest (example different values of k), by visualization. The joint distribution we use in the empirical studies that ensue the theory, is a multinomial — the most generic data generation model for the discrete case.

The paper is organized as follows: In Section 2, we first provide the basic technical background. This is followed by a brief discussion of the kNN algorithm and the different distance metrics that are used when dealing with categorical attributes. In Section 3 we characterize the kNN algorithm. In Section 4 we address issues related to scalability. In Section 5 we report empirical studies, which illustrate the usage of the

Symbol	Meaning
X	Random vector modeling input
$\mathcal{X}$	Domain of random vector (input space) X
Y	Random variable modeling output
$Y(x)$	Random variable modeling output for input x
$\mathcal{Y}$	Set of class labels (output space)
ζ	Classifier
$\mathcal{Z}(N)$	The class of classifiers obtained by application of classification algorithm to an i.i.d. set of size N
$E_{\mathcal{Z}(N)}[]$	Expectation w.r.t. the space of classifiers built on a sample of size N

Table 1 Notation used in the paper.

expressions, as a tool to delve into the statistical behavior of the kNN algorithm. In Section 6, we discuss implications of the experiments. In Section 7, we look at possible extensions to the current work. Finally, we conclude in Section 8.

2 Background

In this section we first discuss the technical framework used in the characterization of the moments. This is followed by a discussion of different distance metrics used by a kNN algorithm when trained on categorical data.

2.1 Technical Framework

In this section we present the generic expressions for the moments of GE that were given in [6]. The moments of the GE of a classifier built over an independent and identically distributed (i.i.d.) random sample drawn from a joint distribution, are taken over the space of all possible classifiers that can be built, given the classification algorithm and the joint distribution. Though the classification algorithm may be deterministic, the classifiers act as random variables since the sample that they are built on is random. The GE of a classifier, being a function of the classifier, also acts as a random variable. Due to this fact, GE of classifier denoted by $GE(\zeta)$ has a distribution and consequently we can talk about its moments. The generic expressions for the first two moments of GE taken over the space of possible classifiers resulting from samples of size N from some joint distribution are as follows:

$$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right] = \sum_{x\in\mathcal{X}} P\left[X=x\right] \sum_{y\in\mathcal{Y}} P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right] P\left[Y(x)\neq y\right] \tag{1}$$

$$\begin{aligned} &E_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[GE(\zeta)GE(\zeta')\right] = \\ &\sum_{x\in\mathcal{X}}\sum_{x'\in\mathcal{X}} P\left[X=x\right] P\left[X=x'\right] \cdot \\ &\quad \sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right] \cdot \\ &\qquad P\left[Y(x)\neq y\right] P\left[Y(x')\neq y'\right] \end{aligned} \tag{2}$$

X	C_1	C_2	...	C_v
$\bar{x_1}$	N_{11}	N_{12}	...	N_{1v}
$\bar{x_2}$	N_{21}	N_{22}	...	N_{2v}
$\vdots$				
$\bar{x_M}$	N_{M1}	N_{M2}	...	N_{Mv}

Table 2 Contingency table with v classes, M input vectors and total sample size $N = \sum_{i=1,j=1}^{M,v} N_{ij}$.

Equation 1 is the expression for the first moment of the $GE(\zeta)$. Notice that inside the first sum $\sum_{x \in \mathcal{X}}$ the input x is fixed and inside the second sum the output y is fixed, thus the $P_{\mathcal{Z}(N)}[\zeta(x)=y]$ is the probability of all possible ways in which an input x is classified into class y. This probability depends on the joint distribution and the classification algorithm. The other two probabilities are directly derived from the distribution. Thus, customizing the expression for $E_{\mathcal{Z}(N)}[GE(\zeta)]$, effectively means deciphering a way of computing $P_{\mathcal{Z}(N)}[\zeta(x)=y]$. Similarly, customizing the expression for $E_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[GE(\zeta)GE(\zeta')\right]$ means finding a way of computing $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ given any joint distribution. In Section 3 we derive expressions for these two probabilities, which depend only on the underlying joint probability distribution, thus providing a way of computing them analytically.

2.2 k-Nearest Neighbor Algorithm

The k-nearest neighbor (kNN) classification algorithm classifies an input based on the class labels of the closest k points in the training dataset. The class label assigned to an input is usually the most numerous class of these k closest points. The underlying intuition that is the basis of this classification model is that nearby points will tend to have higher "similarity" viz. same class, than points that are far apart.

The notion of closeness between points is determined by the distance metric used. When the attributes are continuous, the most popular metric is the l_2 norm or the Euclidean distance. Figure 1 shows points in R^2 space. The points b, c and d are the 3-nearest neighbors (k=3) of the point a. When the attributes are categorical the most popular metric used is the Hamming distance [12]. The Hamming distance between two points/inputs is the number of attributes that have distinct values for the two inputs. This metric is sample independent i.e. the Hamming distance between two inputs remains unchanged, irrespective of the sample counts produced in the corresponding contingency table. For example, Table 2 represents a contingency table. The Hamming distance between $\bar{x_1}$ and $\bar{x_2}$ is the same irrespective of the values of N_{ij} where $i \in \{1, 2, ..., M\}$ and $j \in \{1, 2, ..., v\}$. Other metrics such as Value Difference Metric (VDM) [22], Chi-square [4] etc. exist, that depend on the sample. We now provide a global characterization for calculating the aforementioned probabilities for both kinds of metrics. This is followed by an efficient characterization for the sample independent metrics, which includes the traditionally used and most popular Hamming distance metric.

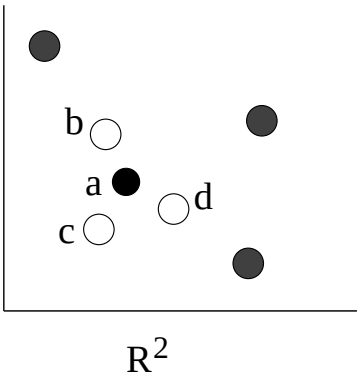


Fig. 1 b, c and d are the 3 nearest neighbours of a.

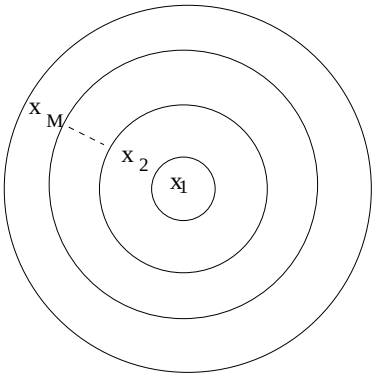


Fig. 2 The Figure shows the extent to which a point $\bar{x_i}$ is near to $\bar{x_1}$. The radius of the smallest encompassing circle for a point $\bar{x_i}$ is proportional to its distance from $\bar{x_1}$. $\bar{x_1}$ is the closest point and $\bar{x_M}$ is the farthest.

3 Computation of Moments

In this section we characterize the probabilities $P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right]$ and $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ required for the computation of the first two moments. In the case, that the number of nearest neighbors at a particular distance d is more than k for an input and at any lesser value of distance the number of NN's is less than k, we classify the input based on all the NN's upto the distance d.

3.1 General Characterization

We provide a global characterization for the above mentioned probabilities without any assumptions on the distance metric in this subsection.

The scenario wherein $\bar{x_i}$ is classified into class C_j given $i \in \{1, 2, ..., M\}$ and $j \in \{1, 2, ..., v\}$ depends on two factors; 1) the kNN's of $\bar{x_i}$ and 2) the class label of the majority of these kNN's. The first factor is determined by the distance metric used, which may be dependent or independent of the sample as previously discussed. The second factor is always determined by the sample. The $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$ is the probability of all possible ways that input $\bar{x_i}$ can be classified into class C_j, given the joint distribution over the input-output space. This probability for $\bar{x_i}$ is calculated by summing the joint probabilities of having a particular set of kNN's and the majority of this set of kNN's has a class label C_j, over all possible kNN's that the input can have. Formally,

$$P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right] = \sum_{q\in Q} P_{\mathcal{Z}(N)}\left[q, c(q,j) > c(q,t), \forall t \in \{1,2,...,v\}, t \neq j\right] \tag{3}$$

where q is a set of kNN's of the given input and Q is the set containing all possible q. $c(q, b)$ is a function which calculates the number of kNN's in q that lie in class

C_b. For example, from Table 2, if $\bar{x_1}$ and $\bar{x_2}$ are the kNN's of some input, then $q = \{1, 2\}$ and $c(q, b) = N_{1b} + N_{2b}$. Notice that, since $\bar{x_1}$ and $\bar{x_2}$ are the kNN's of some input, $\sum_{i=1,j=1}^{2,v} N_{ij} \geq k$. Moreover, if the kNN's comprise of the entire input sample, then the resultant classification is equivalent to classification performed using class priors determined by the sample. The $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ used in the computation of the second moment is calculated by going over kNN's of two inputs rather than one. The expression for this probability is given by,

$$\begin{aligned} &P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j \wedge \zeta'(\bar{x_l})=C_w\right] = \\ &\sum_{q\in Q}\sum_{r\in R} P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[q, c(q,j) > c(q,t), r, c(r,w) > c(r,s) \\ &\qquad \forall s, t \in \{1, 2, ..., v\}, t \neq j, s \neq w] \end{aligned} \tag{4}$$

where q and r are sets of kNN's of $\bar{x_i}$ and $\bar{x_l}$ respectively. Q and R are sets containing all possible q and r respectively. $c(.,.)$ has the same connotation as before.

As mentioned before the probability of a particular q (or the probability of the joint q, r), depends on the distance metric used. The inputs (e.g. $\bar{x_1}$, $\bar{x_2}, ...$) that are the k nearest neighbors to some given input depend on the sample irrespective of the distance metric i.e. the kNN's of an input depend on the sample even if the distance metric is sample independent. We illustrate this fact by an example.

Example 1 Say, $\bar{x_1}$ and $\bar{x_2}$ are the two closest inputs to $\bar{x_i}$ where $\bar{x_1}$ is closer than $\bar{x_2}$, based on some sample independent distance metric. $\bar{x_1}$ and $\bar{x_2}$ are both the kNN's of $\bar{x_i}$ if and only if $\sum_{a=1}^{2}\sum_{b=1}^{v} c(a,b) \geq k$, $\sum_{b=1}^{v} c(\{1\}, b) < k$ and $\sum_{b=1}^{v} c(\{2\}, b) < k$. The first inequality states that the number of copies of $\bar{x_1}$ and $\bar{x_2}$ given by $\sum_{j=1}^{v} N_{1j}$ and $\sum_{j=1}^{v} N_{2j}$ respectively, in the contingency table 2 is greater than or equal to k. If this inequality is true, then definitely the class label of input $\bar{x_i}$ is determined by the copies of $\bar{x_1}$ or $\bar{x_2}$ or both $\bar{x_1}$, $\bar{x_2}$. No input besides these two is involved in the classification of $\bar{x_i}$. The second and third inequality state that the number of copies of $\bar{x_1}$ and $\bar{x_2}$ is less than k respectively. This forces both $\bar{x_1}$ and $\bar{x_2}$ to be used in the classification of $\bar{x_i}$. If the first inequality was untrue, then farther away inputs will also play a part in the classification of $\bar{x_i}$. Thus the kNN's of an input depend on the sample irrespective of the distance metric used.

The above example also illustrates the manner in which the set q (or r) can be characterized as a function of the sample, enabling us to compute the two probabilities required for the computation of the moments from any given joint distribution over the data, for sample independent metrics. Without loss of generality (w.l.o.g.) assume $\bar{x_1}, \bar{x_2}, ..., \bar{x_M}$ are inputs in non-decreasing order (from left to right) of their distance from a given input $\bar{x_i}$ based on some sample independent distance metric. Then this input having the kNN given by the set $q = \{\bar{x_{a1}}, \bar{x_{a2}}, ..., \bar{x_{az}}\}$ where $a1, a2, ..., az \in \{1, 2, ..., M\}$ and $ad < af$ if $d < f$ is equivalent to the following conditions on the sample being true: $\sum_{l=1}^{z}\sum_{j=1}^{v} c(\{\bar{x_{al}}\}, j) \geq k$, $\forall l \in \{1, 2, ..., z\}$ $\sum_{j=1}^{v} c(\{\bar{x_{al}}\}, j) > 0$, $\forall l \in 2^q$ where 2^q is the power set of q and cardinality of $l = z-1$ denoted by $|l| = z-1$, $\sum_{j=1}^{v} c(l, j) < k$ and $\forall \bar{x_h} \in \{\bar{x_1}, \bar{x_2}, ..., \bar{x_M}\} - q$ where $h < az$ $\sum_{j=1}^{v} c(\{\bar{x_h}\}, j) = 0$. The conditions imply that for the elements of q to be the kNN's of some given input, the sum of their counts should be greater than or equal to k, the sum of any subset of the counts (check only subsets of q of cardinality $|q| - 1$) should be less than k, the

count of each element of q is non-zero and the other inputs that are not in q, but are no farther to the given input than the farthest input in q should have counts zero. Notice that all of these conditions are functions of the sample.

The other condition in the probabilities is that a particular class label is the most numerous among the kNNs, which is also a function of the sample. In case of sample dependent metrics, the conditions that are equivalent to having a particular set q as kNN, are totally dependent on the specific distance metric used. Since these distance metrics are sample dependent, we can certainly write these conditions as the corresponding functions of the sample. Since all the involved conditions in the above probabilities can be expressed as functions of the sample, we can be compute them over any joint distribution defined over the data.

3.2 Efficient Characterization for Sample Independent Distance Metrics

In the previous subsection we observed the global characterization for the kNN algorithm. Though this characterization provides insight into the relationship between the moments of GE, the underlying distribution and the kNN classification algorithm, it is inefficient to compute in practical scenarios. This is due to fact that any given input can have itself and/or any of the other inputs as kNN. Hence, the total number of terms in finding the probabilities in equations 3 and 4 turns out to be exponential in the input size M. Considering these limitations, we provide alternative expressions for computing these probabilities efficiently for sample independent distance metrics, viz. Manhattan distance [11], Chebyshev distance [1], Hamming distance. The number of terms in the new characterization we propose, is linear in M for $P_{\mathcal{Z}(N)}[\zeta(x)=y]$ and quadratic in M for $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\zeta(x)=y \wedge \zeta'(x')=y']$.

The characterization we just presented, computes the probability of classifying an input into a particular class for each possible set of kNN's separately. What if we in some manner, combine disjoint sets of these probabilities into groups, and compute a single probability for each group ? This would reduce the number of terms to be computed, thus speeding up the process of computation of the moments. To accomplish this, we use the fact that the distance between inputs is independent of the sample. A consequence of this independence is that all pairwise distances between the inputs are known prior to the computation of the probabilities. This assists in obtaining a sorted ordering of inputs from the closest to the farthest for any given input. For example, if we have inputs a_1b_1, a_1b_2, a_2b_1 and a_2b_2, then given input a_1b_1, we know that a_2b_2 is the farthest from the given input, followed by a_1b_2, a_2b_1 which are equidistant and a_1b_1 is the closest in terms of Hamming distance.

Before presenting a full-fledged characterization for computing the two probabilities, we explain the basic grouping scheme that we employ with the help of an example.

Example 2 W.l.o.g. let $\bar{x_1}$ be the given input, for which we want to find $P_{\mathcal{Z}(N)}[\zeta(\bar{x_1})=C_1]$. Let $\bar{x_1}$, $\bar{x_2}$,...,$\bar{x_M}$ be inputs arranged in increasing order of distance from left to right. This is shown in Figure 2. In this case, the number of terms we need, to compute $P_{\mathcal{Z}(N)}[\zeta(\bar{x_1})=C_1]$ is M. The first term calculates the probability of classifying $\bar{x_1}$ into C_1 when the kNN are multiple instances of $\bar{x_1}$ (i.e. $\sum_{j=1}^{v} N_{1j} \geq k$). Thus, the first group contains only the set $\{\bar{x_1}\}$. The second term calculates the probability of classifying $\bar{x_1}$ into C_1 when the kNN are multiple instances of $\bar{x_2}$ or $\bar{x_1}, \bar{x_2}$. The second group thus contains the sets $\{\bar{x_2}\}$ and $\{\bar{x_1}, \bar{x_2}\}$ as the possible kNN's to $\bar{x_1}$. If we proceed in

this manner, eventually we have M terms and consequently M groups. The M^{th} group will contain sets in which $x_{\bar{M}}$ is an element of every set and the other elements in different sets are all possible combinations of the remaining $M-1$ inputs. Notice that this grouping scheme covers all possible kNN's as in the general case, stated previously i.e. $\cup_{i=1}^{M} g_i = 2^S - \phi$ where g_i denotes the i^{th} group, $S = \{\bar{x_1}, \bar{x_2}, ..., \bar{x_M}\}$, ϕ is the empty set and any two groups are disjoint i.e. $\forall\ i, j \in \{1, 2, ..., M\},\ i \neq j\ g_i \cap g_j = \phi$ preventing multiple computations of the same probability. The r^{th} $(r > 1)$ term in the expression for $P_{\mathcal{Z}(N)}\,[\zeta(\bar{x_1}) = C_1]$ given the contingency table 2 is,

$$
\begin{aligned}
&P_{\mathcal{Z}(N)}\,[\zeta(\bar{x_1}) = C_1, \text{sets in } g_r \in kNN] = \\
&P_{\mathcal{Z}(N)}[\sum_{i=1}^{r} N_{i1} \geq \sum_{i=1}^{r} N_{ij}, \forall\ j \in \{2, 3, ..., v\}, \\
&\qquad \sum_{i=1}^{r}\sum_{l=1}^{v} N_{il} \geq k, \sum_{i=1}^{r-1}\sum_{l=1}^{v} N_{il} < k]
\end{aligned}
\tag{5}
$$

where the last two conditions force only sets in g_r to be among the kNN's. The first condition ensures that C_1 is the most numerous class among the given kNN's. For $r = 1$ the last condition becomes invalid and unnecessary, hence it is removed. The probability for the second moment is the sum of probabilities which are calculated for two inputs rather than one and over two groups, one for each input. W.l.o.g. assume that $\bar{x_1}$ and $\bar{x_2}$ are 2 inputs with $\bar{x_1}$ being the closest input of $\bar{x_2}$ and sets in $g_r \in kNN^{(1)}$ i.e. kNN's for input $\bar{x_1}$ and sets in $g_s \in kNN^{(2)}$ i.e. kNN's for input $\bar{x_2}$ where $r, s \in \{2, 3, ..., M\}$ then the rs^{th} term in $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\,[\zeta(\bar{x_1}) = C_1, \zeta(\bar{x_2}) = C_2]$ is,

$$
\begin{aligned}
&P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\zeta(\bar{x_1}) = C_1, \text{sets in } g_r \in kNN^{(1)}, \\
&\qquad\qquad \zeta(\bar{x_2}) = C_2, \text{sets in } g_s \in kNN^{(2)}] = \\
&P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\sum_{i=1}^{r} N_{i1} \geq \sum_{i=1}^{r} N_{ij}, \forall\ j \in \{2, 3, ..., v\}, \\
&\qquad \sum_{i=1}^{r}\sum_{l=1}^{v} N_{il} \geq k, \sum_{i=1}^{r-1}\sum_{l=1}^{v} N_{il} < k, \\
&\qquad \sum_{i=1}^{s} N_{i2} \geq \sum_{i=1}^{s} N_{ij}, \forall\ j \in \{1, 3, ..., v\}, \\
&\qquad \sum_{i=1}^{s}\sum_{l=1}^{v} N_{il} \geq k, \sum_{i=1}^{s-1}\sum_{l=1}^{v} N_{il} < k]
\end{aligned}
\tag{6}
$$

In this case, when $r = 1$ remove $\sum_{i=1}^{r-1}\sum_{l=1}^{v} N_{il} < k$ condition from the above probability. If $s = 1$ remove $\sum_{i=1}^{s-1}\sum_{l=1}^{v} N_{il} < k$ condition from the above probability.

In the general case, there may be multiple inputs that lie at a particular distance from any given input; i.e. the concentric circles in Figure 2 may contain more than one input. To accommodate this case, we extend the grouping scheme previously outlined. Previously, the group g_r contained all possible sets formed by the $r-1$ distinct closest

inputs to a given input, with the r^{th} closest input being present in every set. Realize that the r^{th} closest input doesn't necessarily mean it is the r^{th} NN, since there may be multiple copies of any of the $r-1$ closest inputs. In our modified definition, the group g_r contains all possible sets formed by the $r-1$ closest inputs, with *at least one* of the r^{th} closest inputs being present in every set. We illustrate this with an example. Say, we have inputs a_1b_1, a_1b_2, a_2b_1 and a_2b_2, then given input a_1b_1, we know that a_2b_2 is the farthest from the given input, followed by a_1b_2, a_2b_1 which are equidistant and a_1b_1 is the closest in terms of Hamming distance. The group g_1 contains only a_1b_1 as before. The group g_2 in this case contains the sets $\{a_1b_2\}$, $\{a_2b_1\}$, $\{a_1b_2, a_2b_1\}$, $\{a_1b_2, a_1b_1\}$ and $\{a_2b_1, a_1b_1\}$. Observe that each set has at least one of the 2 inputs a_1b_2, a_2b_1. We now characterize the probabilities in equations 5 and 6 for this general case. Let q_r denote the set containing inputs from the closest to the r^{th} closest, to some input $\bar{x_i}$. The function $c(.,.)$ has the same connotation as before. With this the r^{th} term in the $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$ where $r \in \{2,3,...,G\}$ and $G \leq M$ is number of groups is,

$$\begin{aligned} &P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j, \text{sets in } g_r \in kNN\right] = \\ &P_{\mathcal{Z}(N)}[c(q_r,j) > c(q_r,l), \forall\ l \in \{1,2,...,v\}\ l \neq j, \\ &\qquad \sum_{t=1}^{v} c(q_r,t) \geq k, \sum_{t=1}^{v} c(q_{r-1},t) < k] \end{aligned} \tag{7}$$

where the last condition is removed for $r=1$. Similarly, the rs^{th} term in $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w\right]$ where $r,s \in \{2,3,...,G\}$ is,

$$\begin{aligned} &P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\zeta(\bar{x_i})=C_j, \text{sets in } g_r \in kNN^{(i)}, \\ &\qquad \zeta(\bar{x_p})=C_w, \text{sets in } g_s \in kNN^{(p)}] = \\ &P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[c(q_r,j) > c(q_r,l), \forall\ l \in \{1,2,...,v\}\ l \neq j, \\ &\qquad \sum_{t=1}^{v} c(q_r,t) \geq k, \sum_{t=1}^{v} c(q_{r-1},t) < k, \\ &\qquad c(q_s,w) > c(q_s,l), \forall\ l \in \{1,2,...,v\}\ l \neq w, \\ &\qquad \sum_{t=1}^{v} c(q_s,t) \geq k, \sum_{t=1}^{v} c(q_{s-1},t) < k] \end{aligned} \tag{8}$$

where $\sum_{t=1}^{v} c(q_{r-1},t) < k$ and $\sum_{t=1}^{v} c(q_{s-1},t) < k$ are removed when $r=1$ and $s=1$ respectively. From equation 7 the $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$ is given by,

$$P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right] = \sum_{r=1}^{G} T_r \tag{9}$$

where T_r is the r^{th} term in $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$. From equation 8 the $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w\right]$ is given by,

$$P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w\right] = \sum_{r,s=1}^{G} T_{rs} \tag{10}$$

where T_{rs} is the rs^{th} term in $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w\right]$.

With this grouping scheme we have been able to reduce the number of terms in the calculation of $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$ and $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w]$ from exponential in M (the number of distinct inputs), to manageable proportions of $\Omega(M)$ terms for the first probability and $\Omega(M^2)$ terms for the second probability. Moreover, we have accomplished this without compromising on the accuracy.

4 Scalability Issues

In the previous section we provided the generic characterization and the time efficient characterization for sample independent distance metrics, relating the two probabilities required for the computation of the first and second moments, to probabilities that can be computed using the joint distribution. In this section we discuss approximation schemes that may be carried out, to further speed up the computation. There are two factors on which the time complexity of calculating $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x_i})=C_j\right]$ and $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\zeta(\bar{x_i})=C_j, \zeta'(\bar{x_p})=C_w]$ depends,

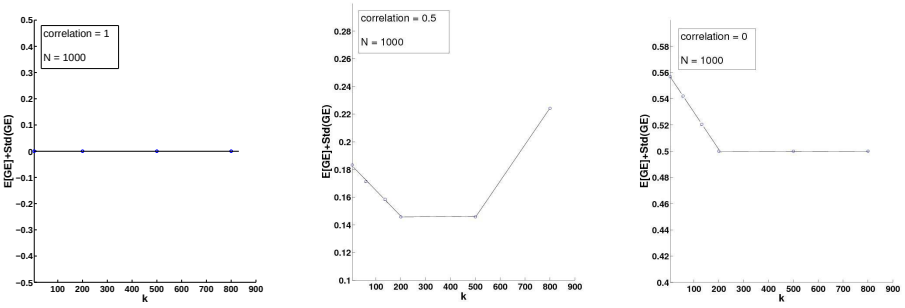
1. the number of terms (or smaller probabilities) that sum up to the above probabilities,
2. the time complexity of each term.

Reduction in number of terms: In the previous section we reduced the number of terms to a small polynomial in M for a class of distance metrics. The current enhancement we propose, further reduces the number of terms and works even for the general case at the expense of accuracy, which we can control. The r^{th} term in the characterizations has the condition that the number of the closest $r-1$ distinct inputs is less than k. The probability of this condition being true monotonically reduces with increasing r. After a point, this probability may become "small enough", so that the total contribution of the remaining terms in the sum is not worthwhile finding, given the additional computational cost. We can set a threshold below which, if the probability of this condition diminishes, we avoid computing the terms that follow.

Reduction in term computation: Each of the terms can be computed directly from the underlying joint distribution. Different tricks can be employed to speed up the computation such as collapsing cells of the table etc., but even then the complexity is still a small polynomial in N. For example, using a multinomial joint distribution, the time complexity of calculating a term for the probability of the first moment is quartic in N and for the probability of the second moment it is octic in N. This problem can be addressed by using the approximation techniques proposed in [6]. Using techniques such as optimization, we can find tight lower and upper bounds for the terms in essentially constant time.

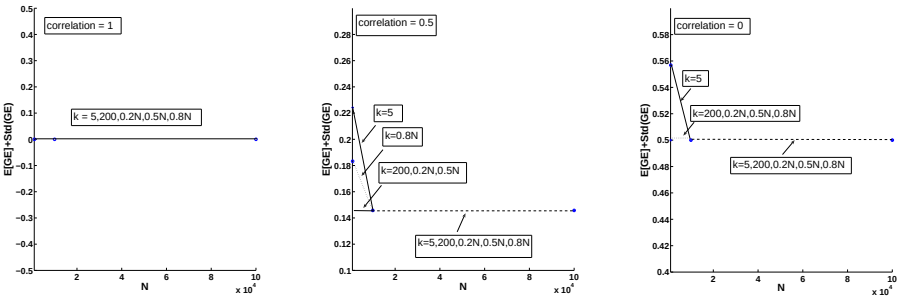
Parallel computation: Note that each of the terms is self contained and not dependent on the others. This fact can be used to compute these terms in parallel, eventually merging them to produce the result. This will further reduce the time of computation.

With this we have not only proposed analytical expressions for the moments of GE for the kNN classification model applied to categorical attributes, but have also suggested efficient methods of computing them.



(a) (b) (c)

Fig. 3 Behavior of the GE for different values of k with sample size $N = 1000$ and the correlation between the attributes and class labels being 1 in (a), 0.5 in (b) and 0 in (c). Std() denotes standard deviation.



(a) (b) (c)

Fig. 4 Convergence of the GE for different values of k when the sample size (N) increases from 1000 to 100000 and the correlation between the attributes and class labels is 1 in (a), 0.5 in (b) and 0 in (c). Std() denotes standard deviation. In (b) and (c), after about $N = 1500$ large, mid-range and small values of k give the same error depicted by the dashed line.

5 Experiments

In this section we portray the manner in which the characterizations can be used to study the kNN algorithm in conjunction with the model selection measures (viz. cross-validation). Generic relationships between the moments of GE and moments of CE (cross-validation error) that are not algorithm specific are given in [6]. We use the expressions provided in this paper and these relationships to conduct the experiments described below. The main objective of the experiments we report, is to provide a flavor of the utility of the expressions as a tool to study this learning method.

5.1 General Setup

We mainly conduct 4 studies. The first three studies are on synthetic data and the fourth on 5 real datasets. In our first study, we observe the performance of the kNN algorithm for different values of k. In the second study, we observe the convergence

behavior of the algorithm with increasing sample size. In our third study, we observe the relative performance of cross-validation in estimating the GE for different values of k. In the three studies we vary the correlation (measured using Chi-square [4]) between the attributes and the class labels to see the effect it has on the performance of the algorithm. In our fourth study, we choose 2 UCI datasets and 3 real industrial datasets, and observe the estimates of cross-validation with the true error estimates. We also explain how a multinomial distribution can be built over these datasets. The same idea can be used to build a multinomial over any discrete dataset to represent it precisely.

Setup for studies 1-3: We set the dimensionality of the space to be 8. The number of classes is fixed to two, with each attribute taking two values. This gives rise to a multnomial with $2^9 = 512$ cells. If we fix the probability of observing a datapoint in cell i to be p_i such that $\sum_{i=1}^{512} p_i = 1$ and the sample size to N, we then have a completely specified multinomial distribution with parameters N and the set of cell probabilities $\{p_1, p_2, ..., p_{512}\}$. The distance metric we use is Hamming distance and the class prior is 0.5.

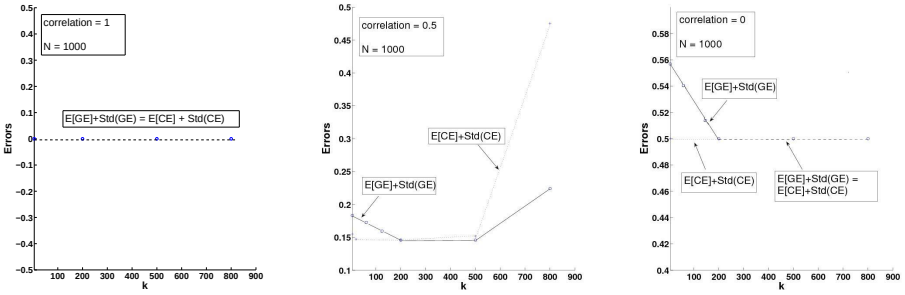
Setup for study 4: In case of real data, we choose 2 UCI datasets whose attributes are *not* limited to having binary splits. One of the proprietary industrial datasets is a supply chain dataset, while the others are an automobile dataset and an invoice amounts dataset.

The supply chain dataset is obtained from an actual manufacturer and contains data at two levels namely; at the (finer) distribution center (DC) level and at the (coarser) manufacturer level. Using the data for a particular DC we predict its inventory position as low, sufficient or excess given past inventory positions and other attributes such as age of the inventory and product type (viz. egg beaters, pasta etc.). The data was collected daily for about a year and hence, the dataset size is 357.

The automobile dataset is obtained from a car manufacturer. This dataset has (quantized) information about the amount of power that a car uses every one tenth of a second, the battery current, the battery voltage, the throttle position, the brake torque and the speed of the car at the corresponding instance. The dataset size is 28127. The goal is to predict the time instants when the (hybrid) car will run on gas and the time instants when it will run on electricity.

The invoice amounts dataset is obtained from a large tech company. Corporations have many of business units (BUs) viz. travel, marketing, auditing, information technology etc. with each BU consisting of various commodity counsils (CCs) viz. office supplies, tech services, communication services, etc. Throughout a calendar year each of the commodity counsils carry out multiple transactions and record the corresponding invoices. It is extremely useful for these CCs and BUs at large, to estimate in advance the invoice amounts that are likely to be registered in the near future. This dataset has 8 BUs with each BU consisting of 150 CCs, leading to a total of $150 \times 8 = 1200$ attributes. The data was collected daily for a year and hence, the dataset has only 365 datapoints. Our target is the CC communication services under the BU information technology which has one of the highest invoice amounts and hence, is critical to business.

All the above datasets can be represented in the form of a contingency table where each cell in the table contains the count of the number of copies of the corresponding input belonging to a particular class. These counts in the individual cells divided by



(a) (b) (c)

Fig. 5 Comparison between the GE and 10 fold Cross validation error (CE) estimate for different values of k when the sample size (N) is 1000 and the correlation between the attributes and class labels is 1 in (a), 0.5 in (b) and 0 in (c). Std() denotes standard deviation.

the dataset size provide us with empirical estimates for the individual cell probabilities (p_i's). Thus, with the knowledge of N (dataset size) and the individual p_i's we have a multinomial distribution whose representative sample is the particular dataset. Using this distribution we observe the estimates of the true error (i.e. moments of GE) and estimates given by cross-validation for different values of k. Notice that these estimates are also applicable (with high probability) to other datasets that are similar to the original.

A detailed explanation of these 4 studies is given below. The expressions in equations 9, 10 are used to produce the plots.

5.2 Study 1: Performance of the kNN algorithm for different values of k.

In the first study we observe the behavior of the GE of the kNN algorithm for different values of k and for a sample size of 1000.

In Figure 3a the attributes and the class labels are totally correlated (i.e. correlation = 1). We observe that for a large range of values of k (from small to large) the error is zero. This is expected since any input lies only in a single class with the probability of lying in the other class being zero.

In Figure 3b we reduce the correlation between the attributes and class labels from being totally correlated to a correlation of 0.5. We observe that for low values of k the error is high, it then plummets to about 0.14 and increases again for large values of k. The high error for low values of k is because the variance of GE is large for these low values. The reason for the variance being large is that the number of points used to classify a given input is relatively small. As the value of k increases this effect reduces upto a stage and then remains constant. This produces the middle portion of the graph where the GE is the smallest. In the right portion of the graph i.e. at very high values of k, almost the entire sample is used to classify any given input. This procedure is effectively equivalent to classifying inputs based on class priors. In the general setup we mentioned that we set the priors to 0.5, which results in the high errors.

In Figure 3c we reduce the correlation still further down to 0 i.e. the attributes and the class labels are uncorrelated. In here we observe that the error is initially high,

then reduces and remains unchanged. As before the initial upsurge is due to the fact that the variance for low values of k is high, which later settles down.

From the three figures, Figure 3a, Figure 3b and Figure 3c we observe a gradual increase in GE as the correlation reduces. The values of k that give low error for the three values correlation and a sample size of 1000 can be deciphered from the corresponding figures. In Figure 3a, we notice that small, mid-range and large values of k are all acceptable. In Figure 3b we find that mid-range values (200 to 500) of k are desirable. In the third figure, i.e. Figure 3c we discover that mid-range and large values of k produce low error.

5.3 Study 2: Convergence of the kNN algorithm with increasing sample size.

In the second study we observe the convergence characteristics of the GE of the kNN algorithm for different values of k, and with increasing sample size going from 1000 to 100000.

In Figure 4a the attributes and class labels are completely correlated. The error remains zero for small, medium and large values of k irrespective of the sample size. In this case any value of k is suitable.

In Figure 4b the correlation between the attributes and the class labels is 0.5. For small sample sizes (less and close to 1000), large and small values of k result in high error while moderate values of k have low error throughout. The initial high error for low values of k is because the variance of the estimates is high. The reason for high error at large values of k is because it is equivalent to classifying inputs based on priors and the prior is 0.5. At moderate values of k both these effects are diminished and hence the error produced by them is low. From the figure we see that after around 1500 the errors of the low and high k converge to the error of moderate k's. Thus here a k within the range 200 to 0.5N would be appropriate.

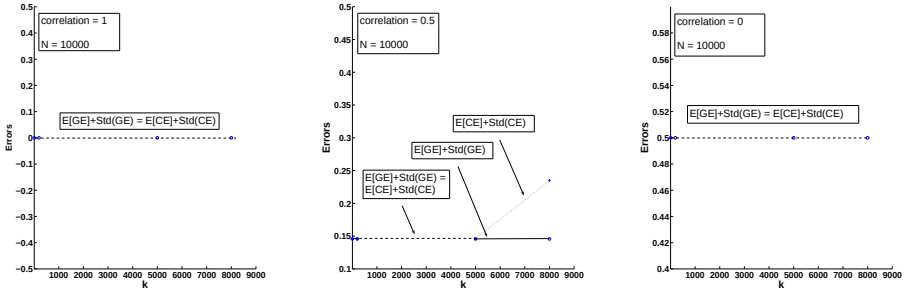
In Figure 4c the attributes and the class labels are uncorrelated. The initial high error for low k's is again because of the high variance. Since the attributes and class labels are uncorrelated with a given prior, the error is 0.5 for moderate as well as high values of k. Here large values of k don't have higher error than the mid-range values since the prior is 0.5. The low value of k converges to the errors of the comparatively larger values at around a sample size of 1500.

Here too from the three figures, Figure 4a, Figure 4b and Figure 4c we observe a gradual increase in GE as the correlation reduces. At sample sizes of greater than about 1500, large, medium and small values of k all perform equally well.

5.4 Study 3: Relative performance of 10-fold cross validation on synthetic data.

In the third and final study on synthetic data we observe the performance of 10-fold Cross validation in estimating the GE for different values of k and sample sizes of 1000 and 10000. The plots for the moments of cross validation error (CE) are produced using the expressions we derived and the relationships between the moments of GE and the moments of CE for deterministic classification algorithms given in [6].

In Figure 5a the correlation is 1 and the sample size is 1000. Cross validation exactly estimates the GE which is zero irrespective of the value of k. When we increase the



(a) (b) (c)

Fig. 6 Comparison between the GE and 10 fold Cross validation error (CE) estimate for different values of k when the sample size (N) is 10000 and the correlation between the attributes and class labels is 1 in (a), 0.5 in (b) and 0 in (c). Std() denotes standard deviation.

sample size to 10000, as shown in Figure 6a Cross validation still does a pretty good job in estimating the actual error (i.e. GE) of kNN.

In Figure 5b the correlation is set to 0.5 and the sample size is 1000. We observe that cross validation initially, i.e. for low values of k underestimates the actual error, performs well for moderate values of k and grossly overestimates the actual error for large values of k. At low values of k the actual error is high because of the high variance, which we have previously discussed. Hence, even-though the expected values of GE and CE are close-by, the variances are far apart, since the variance of CE is low. This leads to the optimistic estimate made by cross validation. At moderate values of k the variance of GE is reduced and hence cross validation produces an accurate estimate. When k takes large values most of the sample is used to classify an input, which is equivalent to classification based on priors. The effect of this is more pronounced in the case of CE than GE, since a higher percentage of the training sample ($\frac{9}{10^{th}}N$) is used for classification of an input for a fixed k, than it is when computing GE. Due to this, CE rises more steeply than GE. When we increase the sample size to 10000, as is depicted in Figure 6b, the poor estimate at low values of k that we saw for a smaller sample size of 1000 vanishes. The reason for this is that the variance of GE reduces with the increase in sample size. Even for moderate values of k the performance of cross validation improves though the difference in accuracy of estimation is not as vivid as in the previous case. For large values of k though the error in estimation is somewhat reduced it is still noticeable. It is advisable that in the scenario presented we should use moderate values of k ranging from about 200 to 0.5N to achieve reasonable amount of accuracy in the prediction made by cross-validation.

In Figure 5c the attributes are uncorrelated to the class labels and the sample size is 1000. For low values of k the variance of GE is high while the variance of CE is low and hence, the estimate of cross validation is off. For medium and large values of k, cross validation estimates the GE accurately, which has the same reason mentioned above. On increasing the sample size to 10000, shown in Figure 6c the variance of GE for low values of k reduces and cross validation estimates the GE with high precision. In general, the GE for any value of k will be estimated accurately by cross validation in this case, but for lower sample sizes (below and around 1000) the estimates are accurate for moderate and large values of k.

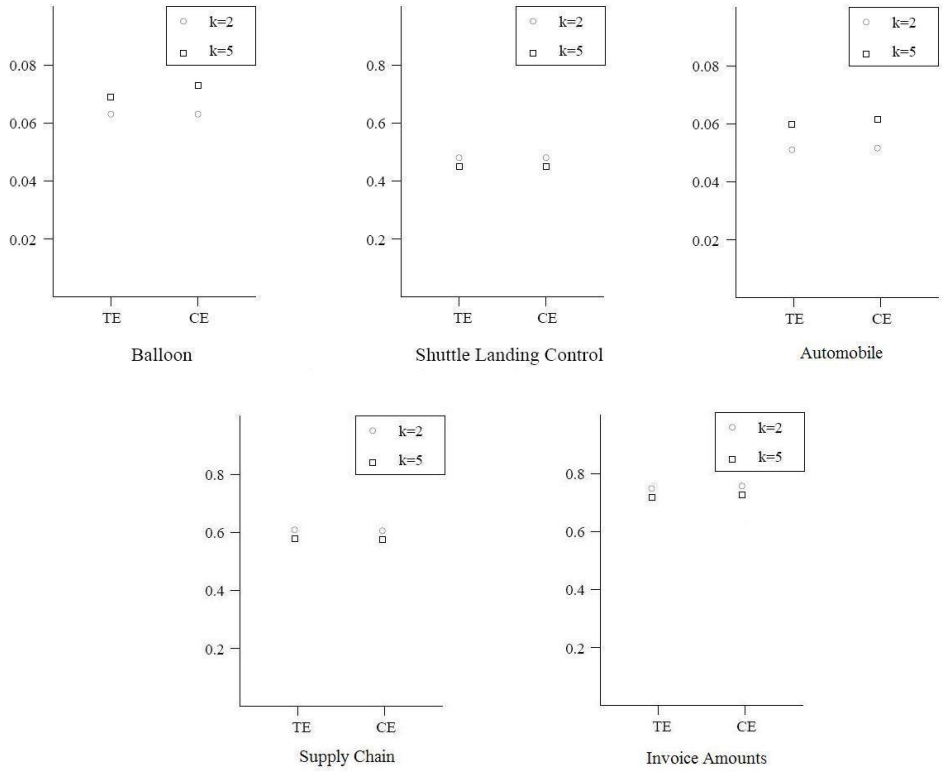


Fig. 7 Comparison between true error (TE) and CE on 2 UCI datasets and 3 industrial datasets obtained from diverse domains.

5.5 Study 4: Relative performance of 10-fold cross validation on real datasets.

In our fourth and final study we observe the behavior of the true error ($E[GE] + Std(GE)$) and the error estimated by cross-validation on 2 UCI datasets. On the Balloon dataset and the Automobile dataset in figure 7 we observe that cross-validation estimates the true error accurately for a k value of 2. Increasing the k to 5 the cross-validation estimate becomes pessimistic. This is because of the increase in variance of CE. We also observe that the true error is lower for k equal to 2. The reason for this is the fact that the expected error is much lower for this case than that for k equal to 5, even-though the variance for k equal to 2 is comparatively higher.

For the Shuttle Landing Control dataset, Supply Chain dataset and Invoice Amounts dataset in figure 7, cross-validation does a good job for both, the small value of k and the larger value of k. The true error in this case is lower for the higher k since the expectations for both the k's is roughly the same but the variance for the smaller k is larger. This is mainly due to the high covariance between the successive runs of cross-validation.

6 Discussion

From the previous section we see that the expressions for the moments assist in providing highly detailed explanations of the observed behavior. Mid-range values of k were the best in studies 1 and 2 for small sample sizes. The reason for this is the fact that at small values of k the prediction was based on individual cells and having a small sample size the estimates were unstable, producing a large variance. For high values of k the classification was essentially based on class priors and hence the expected error was high, even-though the variance in this case was low. In the case of mid-range values of k, the pitfalls of the extreme values of k were circumvented (since k was large enough to reduce variance but small enough so as to prevent classification based on priors) and hence the performance was superior. 10-fold cross-validation which is considered to be the "holy-grail" in error estimation, is not always ideal as we have seen in the experiments. The most common reason why cross-validation under performed in certain specific cases, was that its variance was high, which in turn was due to the covariance between successive runs of cross-validation was high. The ability to make such subtle observations and provide meticulous explanations for them, is the key strength of the deployed methodology – developing and using the expressions.

Another important aspect is that, in the experiments, we built a single distribution on each test dataset to observe the best value of k. Considering the fact that data can be noisy we can build multiple distributions with small perturbations in parameters (depending on the level of noise) and observe the performance of the algorithm for different values of k using the expressions. Then we can choose a robust value of k for which the estimates of the error are acceptable on most (or all) built distributions. Notice that this value of k may not be the best choice, on the distribution built without perturbations. We can thus use the expressions to make these type of informed decisions.

As we can see, by building expressions for the moments of GE in the manner portrayed, classification models in conjunction with popular model selection measures can be studied in detail. The expressions can act as a guiding tool in making the appropriate choice of model and model selection measure in desired scenarios. For example, in the experiments we observed that 10-fold cross-validation did not perform well in certain cases. In these cases we can use the expressions to study cross-validation with different number of folds and attempt to find the ideal number of folds for our specific situation. Moreover, such characterizations can aid in finding answers or challenging the appropriateness of questions such as, *What number of v in v-fold cross-validation gives the best bias/variance trade-off?*. The appropriateness of some queries has to be sometimes challenged since it may very well be the case that no single value of v is truly optimal. In fact depending on the situation, different values of v or may be even other model selection measures (viz. hold out set etc.) may be optimal. Analyzing such situations and finding the appropriate values of the parameters (i.e. v for cross-validation, f–fraction of hold-out, for hold out set validation) can be accomplished using the methodology we have deployed in the paper. Sometimes, it is intuitive to anticipate the behavior of a learning algorithm in extreme cases, but the behavior at the non-extreme cases is not as intuitive. Moreover, the precise point at which the behavior of an algorithm starts to emulate the particular extreme case is a non-trivial task. The methodology can be used to study such cases and potentially a wide range of other relevant questions. Essentially, the studies 1 and 2 in the experimental section are examples of such studies.

In those experiments, at extreme correlations the behavior is more or less predictable but at intermediate correlations it is not.

What the studies in the experimental section and the discussion above suggest is that the method in [6] and developments such as the ones introduced in this paper open new avenues in studying learning methods, allowing them to be assessed for their robustness, appropriateness for a specific task, with lucid elucidations being given for their behavior. These studies do not replace but complement purely theoretical and empirical studies usually carried out when evaluating learning methods.

7 Possible Extensions

We discussed the importance of the methodology in the previous section. Below, we touch upon ways of extending the analysis provided in this paper. An interesting line of future research would be to efficiently characterize the sample dependent distance metrics. Another interesting line would be to extend the analysis to the continuous kNN classification algorithm. A possible way of doing this would be to consider a set of k points that would be kNN to a given input (recollect that to characterize the moments, we only need to characterize the behavior of the algorithm on individual inputs e.g. $P_{\mathcal{Z}(N)}\left[\zeta(\bar{x}_i)=C_j\right]$) and consider the remaining N-k points to lie outside the smallest ball encompassing the kNN. Under these conditions we would integrate the density defined on the input/output space over all possible such N (i.e. k which are kNN and the remaining N-k) with the appropriate condition for class majority (i.e. to classify an input in C_i, we would have the condition that, at least $\lfloor \frac{k}{v} \rfloor + 1$ points that are kNN lie in class C_i). A rigorous analysis using ideas from this paper would have to be performed and the complexity discussed for the continuous kNN. We plan to address these issues in the future.

8 Conclusion

In this paper, we provided a general characterization for the moments of GE of the kNN algorithm applied to categorical data. In particular, we developed an efficient characterization for the moments when the distance metric was sample independent. We discussed issues related to scalability in using the expressions and suggested optimizations to speedup the computation. We later portrayed the usage of the expressions and hence the methodology with the help of empirical studies. It remains to be seen how extensible such an analysis is, to other learning algorithms. However, if such an analysis is in fact possible, it can be deployed as a tool to better understanding the statistical behavior of learning models in the non-asymptotic regime.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 0448264.

References

1. J. Abello, P. Pardalos, and M. Resende, editors. *Handbook of massive data sets*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.
2. A. Blum, A. Kalai, and J. Langford. Beating the hold-out: Bounds for k-fold and progressive cross-validation. In *Computational Learing Theory*, 1999.
3. D. Bremner, E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin, and G. Toussaint. Output-sensitive algorithms for computing nearest-neighbor decision boundaries. *Discrete and Computational Geometry*, 33:593–604, 2005.
4. J. Connor-Linton. Chi square tutorial. http://www.georgetown.edu/faculty/ballc/webtools/web_chi_tut.html, 2003.
5. A. Dhurandhar and A. Dobra. Probabilistic characterization of random decision trees. *Journal of Machine Learning Research*, 9, 2008.
6. A. Dhurandhar and A. Dobra. Semi-analytical method for analyzing models and model selection measures based on moment analysis. *ACM Transactions on Knowledge Discovery from Data*, 3, March 2009.
7. R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley & Sons, 2 edition, 2001.
8. M. Hall and G. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE TRANSACTIONS ON KDE*, 2003.
9. Q. Hu, W. Pan, S. An, P. Ma, and J. Wei. An efficient gene selection technique for cancer recognition based on neighborhood mutual information. *Int. J. Mach. Learn. & Cyber.*, 1:63–74, 2010.
10. R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *In Proceedings of the Fourteenth IJCAI.*, 1995.
11. E. Krause. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Dover, 1987.
12. W. Liu and A. White. Metrics for nearest neighbour discrimination with categorical attributes. In *Research and Development in Expert Systems XIV: Proceedings of the 17th Annual Technicial Conference of the BCES Specialist Group*, pages 51–59, 1997.
13. M. Maggini, C. Giles, and B. Horne. Financial time series forecasting using k-nearest neighbors classification. In *NONLINEAR FINANCIAL FORECASTING*, pages 169–181, 1997.
14. A. Moore and M. Lee. Efficient algorithms for minimizing cross validation error. In *International Conference on Machine Learning*, pages 190–198, 1994.
15. F. Nigsch, A. Bender, B. Buuren, J. Tissen, E. Nigsch, and J. Mitchell. Melting point prediction employing k-nearest neighbor algorithms and genetic parameter optimization. *Journal of Chemical Information and Modeling*, 46:2412–2422, 2006.
16. B. Park and R. Samworth. Choice of neighbor order in nearest-neighbor classification. *Annals of Statistics*, 36:2135–2152, 2008.
17. Y. Qin, D. Zheng, and T. Zhao. Research on search results optimization technology with category features integration. *Int. J. Mach. Learn. & Cyber.*, 3:71–76, 2012.
18. B. Rajagopalan and U. Lall. A k-nearest neighbor simulator for daily precipitation and other weather variables. *Water Resour. Res*, 35:3089–3101, 1999.
19. J. Shao. *Mathematical statistics*. Springer-Verlag, 2003.
20. G. Sidorov, M. Koeppen, and N. Cruz-Corts. Recent advances in machine learning techniques and applications. *Int. J. Mach. Learn. & Cyber.*, 2:123–124, 2011.
21. J. Sim, S. Kim, and J. Lee. Prediction of protein solvent accessibility using fuzzy k-nearest neighbor method. *Bioinformatics/computer Applications in The Biosciences*, 21:2844–2849, 2005.
22. C. Stanfill and D. Waltz. Toward memory-based reasoning. *Commun. ACM*, 29(12):1213–1228, 1986.
23. C. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 5(4):595–645, 1977.
24. V. Vapnik. *Statistical Learning Theory*. Wiley & Sons, 1998.
25. K. Yang and C. Shahabi. An efficient k nearest neighbor search for multivariate time series. *Inf. Comput.*, 205:65–98, January 2007.