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

Insights into Cross-validation

Amit Dhurandhar · Alin Dobra

Received: date / Accepted: date

Abstract Cross-validation is one of the most widely used techniques, in estimating the Generalization Error of classification algorithms. Though several empirical studies have been conducted, to study the behavior of this method in the past, none of them clearly elucidate the reasons behind the observed behavior. In this paper we study the behavior of the moments (i.e. expected value and variance) of the cross-validation Error and explain the observed behavior in detail. In particular, we provide interesting insights into the behavior of covariance between the individuals runs of cross-validation, which has significant effects on the overall variance. We study this behavior on three classification models which are a mix of parametric and non-parametric models namely, Naive Bayes Classifier – parametric model, Decision Trees and K-Nearest Neighbor Classifier – non-parametric models. The moments are computed using closed form expressions rather than directly using Monte Carlo since the former is shown to be a more viable alternative. The work in this paper complements the prior experimental work done in studying this method since it explains in detail the reasons for the observed trends in the experiments, as opposed to simply reporting the observed behavior.

Keywords Cross-validation · Covariance

1 Introduction

A major portion of the research in Machine Learning is devoted to building classification models. The errors that these models make on the entire input i.e. expected loss over the entire input is defined as their Generalization Error (GE). Ideally, we would want to choose the model with the lowest GE. In practice though, we do not have the entire input and consequently we cannot compute the GE. Nonetheless, a number of methods have been proposed namely, hold-out-set, akaike information criteria, bayes

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

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

information criteria, cross-validation etc. which aim at estimating the GE from the available input.

In this paper we focus on cross-validation, which is arguably one of the most popular GE estimation methods. In v-fold cross-validation the dataset of size N is divided into v equal parts. The classification model is trained on $v-1$ parts and tested on the remaining part. This is performed v times and the average error over the v runs is considered as an estimate of GE. This method is known to have low variance [15,18] for about 10-20 folds and is hence commonly used for small sample sizes.

Most of the experimental work on cross-validation focusses on reporting observations [15,18] and not on understanding the reasons for the observed behavior. Moreover, modeling the covariance between individual runs of cross-validation is not a straightforward task and is hence not adequately studied, though it is considered to have a non-trivial impact on the behavior of cross-validation. The work presented in [1,16] address issues related to covariance, but it is focussed on building and studying the behavior of estimators for the overall variance of cross-validation. In [16] the estimators of the moments of cross-validation error (CE) are primarily studied for the estimation of mean problem and in the regression setting. The goal of this paper is quite different. We do not wish to build estimators for the moments of CE rather we want to experimentally observe the behavior of the moments of cross-validation and provide explanations for the observed behavior in the classification setting. The classification models we run these experiments on consist of the Naive Bayes Classifier (NBC) – a parametric model and two non-parametric models namely, K-Nearest Neighbor Classifier (KNN) and Decision Trees (DT). We choose a mix of parametric and non-parametric models so that their hypothesis spaces are varied enough. Additionally, these models are widely used in practice. The moments however, are computed not using Monte Carlo directly but using the expressions given in [8,7,6] which are also given in the appendix of this paper. The advantage of using these closed form expressions is that they are *exact* formulas (not approximations) for the moments of CE and hence these moments can be studied accurately with respect to any chosen distribution. In fact as it turns out approximating certain probabilities in these expressions also leads to significantly higher accuracy in computing the moments when compared with directly using Monte Carlo. The reason for this is that the parameter space of the individual probabilities that need to be computed in these expressions is much smaller than the space over which the moments have to be computed at large and hence directly using Monte Carlo to estimate the moments can prove to be highly inaccurate in many cases [8, 7]. Another advantage of using the closed form expressions is that they give us more control over the settings we wish to study.

In summary, the goal of the paper is to empirically study the behavior of the moments of CE (plotted using the expressions in the Appendix) and to provide interesting explanations for the observed behavior. As we will see, when studying the variance of CE, the covariance between the individual runs plays a decisive role and hence understanding its behavior is critical in understanding the behavior of the total variance and consequently the behavior of CE. We provide insights into the behavior of the covariance apropos increasing sample size, increasing correlation between the data and the class labels and increasing number of folds.

In the next section we review some basic definitions and previous results that are relevant to the computation of the moments of CE. In Section 3 we provide an overview of the expressions customized to compute the moments of CE for the three classification algorithms namely; DT, NBC and KNN. In Section 4 we conduct a brief literature

survey. In Section 5 – the experimental section, we provide some keen insights into the behavior of cross-validation, which is our primary goal. We discuss the implications of the study conducted and summarize the major developments in the paper in Section 6.

2 Preliminaries

Probability distributions completely characterize the statistical behavior of a random variable. Moments of a random variable give us information about its probability distribution. Thus, if we have knowledge of the moments of a random variable we can make statements about its behavior. In some cases characterizing a finite subset of moments may prove to be a more desired alternative than characterizing the entire distribution which can be computationally expensive. By employing moment analysis and using linearity of expectation efficient generalized expressions for the moments of GE and relationships between the moments of GE and the moments of CE were derived in [8]. In this section we review the relevant results which are used in the present study of CE.

Consider that N points are drawn independently and identically (i.i.d.) from a given distribution and a classification algorithm is trained over these points to produce a classifier. If multiple such sets of N i.i.d. points are sampled and a classification algorithm is trained on each of them we would obtain multiple classifiers. Each of these classifiers would have its own GE, hence the GE is a random variable defined over the space of classifiers which are induced by training a classification algorithm on each of the datasets that are drawn from the given distribution. The moments of GE computed over this space of all possible such datasets of size N, depend on three things: 1) the number of samples N, 2) the particular classification algorithm and 3) the given underlying distribution. We denote by $D(N)$ the space of datasets of size N drawn from a given distribution. The moments taken over this new distribution – the distribution over the space of datasets of a particular size, are related to the moments taken over the original given distribution which is over individual inputs in the following manner,

$$
\begin{aligned}
&E_{D(N)}[\mathcal{F}(\zeta)] \\
&= E_{(\mathcal{X}\times\mathcal{Y})\times(\mathcal{X}\times\mathcal{Y})\times\ldots\times(\mathcal{X}\times\mathcal{Y})}[\mathcal{F}(\zeta)] \\
&= \sum_{(x_1,y_1)\in\mathcal{X}\times\mathcal{Y}} \sum_{(x_2,y_2)\in\mathcal{X}\times\mathcal{Y}} \ldots \sum_{(x_N,y_N)\in\mathcal{X}\times\mathcal{Y}} P[X_1 = x_1 \wedge Y_1 = y_1 \wedge \cdots \wedge X_N = x_N \wedge Y_N = y_N] \cdot \\
&\qquad\qquad \mathcal{F}(\zeta(x_1,\ldots,x_N)) \\
&= \sum_{(x_1,y_1)\in\mathcal{X}\times\mathcal{Y}} \sum_{(x_2,y_2)\in\mathcal{X}\times\mathcal{Y}} \ldots \sum_{(x_N,y_N)\in\mathcal{X}\times\mathcal{Y}} \left(\prod_{i=1}^{N} P[X_i = x_i \wedge Y_i = y_i]\right) \mathcal{F}(\zeta(x_1,\ldots,x_N))
\end{aligned}
$$

where $\mathcal{F}(.)$ is some function that operates on a classifier ζ and ζ is a classifier obtained by training on a particular dataset belonging to $D(N)$. $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output space respectively. $X_1, ..., X_N$ denote a set of N i.i.d. random variables defined over the input space and $Y_1, ..., Y_N$ denote a set of N i.i.d. random variables defined over the output space. For simplicity of notation we denote the moments over the space of the new distribution rather than over the space of the given distribution.

Notice that in the above formula, $E_{D(N)}[\mathcal{F}(\zeta)]$ was expressed in terms of product of probabilities since the independence of samples $x_1, y_1, \ldots, x_n, y_n$ allows for the factorization of $P[X_1 = x_1 \wedge Y_1 = y_1 \wedge \cdots \wedge X_N = x_N \wedge Y_N = y_N]$. By instantiating the function $\mathcal{F}(.)$ with $GE(.)$, we have a formula for computing moments of GE. The problem with this characterization is that it is highly inefficient (exponential in the size of the input-output space). Efficient characterizations for computing the moments were developed in [8] which we will shortly review. The characterization reduces the number of terms in the moments from an exponential in the input-output space to linear for the computation of the first moment and quadratic for the computation of the second moment.

We define random variables of interest namely, hold-out error, cross-validation error and generalization error. We also define moments of the necessary variables. In the notation used to represent these moments we write the probabilistic space over which the moments are taken as subscripts. If no subscript is present, the moments are taken over the original input-output space. This convention is strictly followed in this particular section. In the remaining sections we drop the subscript for the moments since the formulas can become tedious to read and the space over which these moments are computed can be easily deciphered from the context.

Hold-out Error (HE): The hold-out procedure involves randomly partitioning the dataset D into two parts D_t – the training dataset of size N_t and D_s – the test datasets of size N_s. A classifier is built over the training dataset and the error is estimated as the average error over the test dataset. Formally,

$$HE = \frac{1}{N_s} \sum_{(x,y) \in D_s} \lambda(\zeta(x), Y(x))$$

where $Y(x) \in \mathcal{Y}$ is a random variable modeling the true class label of the input x, $\lambda(.,.)$ is a 0-1 loss function which is 0 when its 2 arguments are equal and is 1 otherwise. ζ is the classifier built on the training data D_t with $\zeta(x)$ being its prediction on the input x.

Cross Validation Error (CE): v-fold cross validation consists in randomly partitioning the available data into v chunks and then training v classifiers using all data but one chunk and then testing the performance of the classifier on this chunk. The estimate of the error of the classifier built from the entire data is the average error over the chunks. Denoting by HE_i the hold-out error on the i^{th} chunk of the dataset D, the cross-validation error is given by,

$$CE = \frac{1}{v} \sum_{i=1}^{v} HE_i$$

Generalization Error (GE): The GE of a classifier ζ w.r.t. the underlying distribution over the input-output space $\mathcal{X} \times \mathcal{Y}$ is given by,

$$\begin{aligned} GE(\zeta) &= E\left[\lambda(\zeta(x), Y(x))\right] \\ &= P\left[\zeta(x) \neq Y(x)\right] \end{aligned}$$

where $x \in \mathcal{X}$ is an input and $Y(x) \in \mathcal{Y}$ the true class label of the input x. It is thus the expected error over the entire input.

Moments of GE: Given an underlying input-output distribution and a classification algorithm, by generating N i.i.d. datapoints from the underlying distribution and training the classification algorithm on these datapoints we obtain a classifier ζ. If we sample multiple (in fact all possible) such sets of N datapoints from the given distribution and train the classification algorithm on each of them, we induce a space of classifiers trained on a space of datasets of size N denoted by $D(N)$. Since the process of sampling produces a random sample of N datapoints, the classifier induced by training the classification algorithm on this sample is a random function. The generalization error w.r.t. the underlying distribution of classifier ζ, denoted by $GE(\zeta)$ is also a random variable that can be studied and whose moments are given by,

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

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

where $\mathcal{X}$[1] and $\mathcal{Y}$ denote the space of inputs and outputs respectively. $Y(x)$ represents the true output[2] for a given input x. $P[x]$ and $P\left[x'\right]$ represent the probability of having a particular input. ζ' in equation 2 is a classifier like ζ (may be same or different) induced by the classification algorithm trained on a sample from the underlying distribution. $P_{D(N)}[\zeta(x) = y] \, P[Y(x) \neq y]$ represents the probability of error. The first probability in the product $P_{D(N)}[\zeta(x) = y]$, depends on the classification algorithm and the data distribution that determines the training dataset. The second probability $P[Y(x) \neq y]$, depends only on the underlying distribution. Also note that both these probabilities are actually conditioned on x but we omit writing the probabilities as conditionals explicitly since it is an obvious fact and it makes the formulas more readable. $E_{D(N)}[.]$ denotes the expectation taken over all possible datasets of size N drawn from the data distribution. The terms in equation 2 also have similar semantics but are applicable to pairs of inputs and outputs. Thus, by being able to compute each of these probabilities we can compute the moments of GE.

Moments of CE: The process of sampling a dataset (i.i.d.) of size N from a probability distribution and then partitioning it randomly into two disjoint parts of size N_t and N_s, is statistically equivalent to sampling two different datasets of size N_t and N_s i.i.d. from the same probability distribution. The first moment of CE is just the expected error of the individual runs of cross-validation. In the individual runs the dataset is partitioned into disjoint training and test sets. $D_t(N_t)$ and $D_s(N_s)$ denote the space of training sets of size N_t and test sets of size N_s respectively. Hence, the first moment of CE is taken w.r.t. the $D_t(N_t) \times D_s(N_s)$ space which is equivalent to the space obtained by sampling datasets of size $N = N_t + N_s$ followed by randomly splitting them into

[1] If input is continuous we replace sum over $\mathcal{X}$ by integrals in the above formulas, everything else remaining same.

[2] Y(.) may or may not be randomized.

training and test sets. In the computation of variance of CE we have to compute the covariance between any two runs of cross-validation (equation 3). In the covariance we have to compute the following cross moment, $E_{D_t^{ij}(\frac{v-2}{v}N)\times D_s^i(\frac{N}{v})\times D_s^j(\frac{N}{v})}\left[HE_iHE_j\right]$ where $D_t^{ij}(k)$ is the space of overlapped training sets of size k in the i^{th} and j^{th} run of cross-validation ($i, j \leq v$ and $i \neq j$), $D_s^f(k)$ is the space of all test sets of size k drawn from the data distribution in the f^{th} run of cross-validation ($f \leq v$), $D_t^f(k)$ is the space of all training sets of size k drawn from the data distribution in the f^{th} run of cross-validation ($f \leq v$) and HE_f is the hold-out error of the classifier in the f^{th} run of cross-validation. Since the cross moment considers interaction between two runs of cross-validation it is taken over a space consisting of training and test sets involving both the runs rather than just one. Hence, the subscript in the cross moments is a cross product between 3 spaces (overlapped training sets between two runs and the corresponding test sets). The other moments in the variance of CE are taken over the same space as the expected value. The variance of CE is given by,

$$
\begin{aligned}
Var(CE) &= \frac{1}{v^2}[\sum_{i=1}^{v} Var(HE_i) + \sum_{i,j,i\neq j}^{v} Cov(HE_i, HE_j)] \\
&= \frac{1}{v^2}[\sum_{i=1}^{v}(E_{D_t^i(\frac{v-1}{v}N)\times D_s^i(\frac{N}{v})}[HE_i^2] - E^2_{D_t^i(\frac{v-1}{v}N)\times D_s^i(\frac{N}{v})}[HE_i]) \\
&+ \sum_{i,j,i\neq j}^{v}(E_{D_t^{ij}(\frac{v-2}{v}N)\times D_s^i(\frac{N}{v})\times D_s^j(\frac{N}{v})}\left[HE_iHE_j\right] \\
&- E_{D_t^i(\frac{v-1}{v}N)\times D_s^i(\frac{N}{v})}[HE_i]E_{D_t^j(\frac{v-1}{v}N)\times D_s^j(\frac{N}{v})}[HE_j]]
\end{aligned}
$$

The reason we introduced moments of GE previously is that, in [8] relationships were drawn between these moments and the moments of CE. Thus, using the expressions for the moments of GE and the relationships which we will state shortly, we have expressions for the moments of CE.

The relationship between the expected values of CE and GE is given by,

$$E_{D_t(\frac{v-1}{v}N)\times D_s(\frac{N}{v})}[CE] = E_{D_t(\frac{v-1}{v}N)}[GE(\zeta)]$$

where v is the number of folds, $D_t(k)$ is the space of all training sets of size k that are drawn from the data distribution and $D_s(k)$ is the space of all test sets of size k drawn from the data distribution.

In the computation of variance of CE we need to find the individual variances and the covariances. In [8] it was shown that
$E_{D_t^i(\frac{v-1}{v}N)\times D_s^i(\frac{N}{v})}[HE_i] = E_{D_t(\frac{v-1}{v}N)\times D_s(\frac{N}{v})}[CE]\ \forall i \in \{1, 2, ..., v\}$ and hence the expectation of HE_i can be computed using the above relationship between the expected CE and expected GE. Notice that the space of training and test datasets over which the moments are computed is the same for each fold (since the space depends only on the size and all the folds are of the same size) and hence the corresponding moments are also the same. To compute the remaining terms in the variance we use the following relationships.

The relationship between the second moment of $HE_i\ \forall i \in \{1, 2, ..., v\}$ and the moments of GE is given by,

$$E_{D_t^i(\frac{v-1}{v}N)\times D_s^i(\frac{N}{v})}[HE_i^2] = \frac{v}{N}E_{D_t(\frac{v-1}{v}N)}\left[GE(\zeta)\right] + \frac{N-v}{N}E_{D_t(\frac{v-1}{v}N)}\left[GE(\zeta)^2\right]$$

The relationship between the cross moment and the moments of GE is given by,

$$E_{D_t^{ij}(\frac{v-2}{v}N)\times D_s^i(\frac{N}{v})\times D_s^j(\frac{N}{v})}\left[HE_iHE_j\right] = E_{D_t^{ij}(\frac{v-2}{v}N)}\left[GE(\zeta^i)GE(\zeta^j)\right]$$

where ζ^f is the classifier built in the f^{th} run of cross-validation. All the terms in the variance can be computed using the above relationships.

3 Overview of the customized expressions

In Section 2 we provided the generalized expressions for computing the moments of GE and consequently moments of CE. In particular the moments we compute are: $E[CE]$ and $Var(CE)$. The formula for the variance of CE can be rewritten as a convex combination of the variance of the individual runs and the covariance between any two runs. Formally,

$$Var(CE) = \frac{1}{v}Var(HE) + \frac{v-1}{v}Cov(HE_i, HE_j) \tag{3}$$

where HE is the error of any of the individual runs and HE_i and HE_j are the errors of the i^{th} and j^{th} runs. Since the moments are over all possible datasets the variances and covariances are same for all single runs and pairs of runs respectively and hence the above formula.

In the expressions for the moments the probabilities $P[\zeta(x)=y]$ and $P\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ are the only terms that depend on the particular classification algorithm. Customizing the expressions for the moments equates to finding these probabilities. The other terms in the expressions are straightforward to compute from the data distribution. We now give a high level overview of how the two probabilities are customized for DT, NBC and KNN. The precise details are given in Appendix A.

1. *DT:* To find the probability of classifying an input into a particular class ($P[\zeta(x)=y]$), we have to sum the probabilities over all paths (path is a set of nodes and edges from root to leaf in a tree) that include the input x and have majority of the datapoints lying in them belong to class y, in the set of possible trees. These probabilities can be computed by fixing the split attribute selection method, the stopping criteria deployed to curb the growth of the tree and the data distribution from which an input is drawn. The probability $P\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ can be computed similarly, by considering pairs of paths (one for each input) rather than a single path. The details are given in Appendix A.1.
2. *NBC:* The NBC algorithm assumes that the attributes are independent of each other. An input is classified into a particular class, if the product of the class conditionals of each attribute for the input multiplied by the particular class prior is greater than the same quantity computed for each of the other classes. To compute $P[\zeta(x)=y]$, the probability of the described quantity for input x belonging to class y being greater than the same quantity for each of the other classes is computed. The probability for the second moment can be computed analogously by considering pairs of inputs and outputs. The details are given in Appendix A.2.

3. *KNN:* In KNN to compute $P[\zeta(x)=y]$, the probability that majority of the K nearest neighbors of x belong to class y is found. In the extreme case when K=N (K is the dataset size) the probability of the empirical prior of class y being greater than the empirical priors of the other classes is computed.In this case the majority classifier and KNN behave the same way. To compute $P\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ the probability that the majority of the K nearest neighbors of x belong to class y and majority of the K nearest neighbors of x' belong to class y' is found. The details are given in Appendix A.3.

By using the customized expressions we can accurately study the behavior of cross-validation. This method is more accurate than directly using Monte-Carlo to estimate the moments since the parameter space of the individual probabilities that need to be estimated is much smaller than the entire space over which the moments are computed.

4 Related Work

There is a large body of both experimental and theoretical work that addresses the problem of studying cross validation. In [9] cross validation is studied in the linear regression setting with squared loss and is shown to be biased upwards in estimating the mean of the true error. More recently the same author in [10], compared parametric model selection techniques namely, covariance penalties with the non-parametric cross validation and showed that under appropriate modeling assumptions the former is more efficient than cross validation. [3] showed that cross validation gives an unbiased estimate of the first moment of GE. Though cross validation has desired characteristics with estimating the first moment, Breiman stated that its variance can be significant. In [17] heuristics are proposed to speed up cross-validation which can be an expensive procedure with increasing number of folds. In [23] a simple setting was constructed in which cross-validation performed poorly. [12] refuted this proposed setting and claimed that a realistic scenario in which cross-validation fails is still an open question.

The major theoretical work on cross-validation is aimed at finding bounds. The current distribution free bounds [5,14,2,11,22] for cross-validation are loose with some of them being applicable only in restricted settings such as bounds that rely on algorithmic stability assumptions. Thus, finding tight PAC (Probably Approximately Correct) style bounds for the bias/variance of cross-validation for different values of v is still an open question [13].

Though bounds are useful in their own right they do not aid in studying trends of the random variable in question,[3] in this case CE. Asymptotic analysis can assist in studying trends [21,20] with increasing sample size, but it is not clear when the asymptotics come into play. This is where empirical studies are useful. Most empirical studies on cross-validation indicate that the performance (bias+variance) is the best around 10-20 folds [15,3] while some others [19] indicate that the performance improves with increasing number of folds. In the experimental study that we conduct using closed form expressions, we observe both of these trends but in addition we provide lucid elucidations for observing such behavior.

[3] unless they are extremely tight.

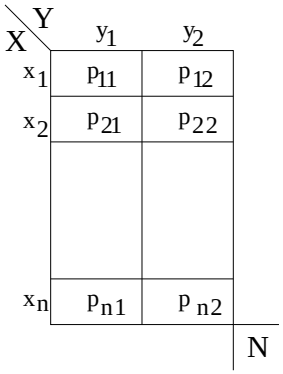

Fig. 1 The above figure depicts the data generation model used in the experiments. X and Y denote the input and output space respectively. p_{ij} where $i \in \{1, ..., n\}$ and $j \in \{1, 2\}$ is the probability of sampling the datapoint (x_i, y_j) such that $\sum_{i=1}^{n} \sum_{j=1}^{2} p_{ij} = 1$. Given a sample size N and these probabilities we have a multinomial data generation model.

5 Experiments

In this section we study cross-validation by explaining its behavior in detail. Especially noticeable, is the behavior of the covariance of cross-validation which plays a significant role in the overall variance. The role of covariance is significant since in the expression for the overall variance given by $Var(CE) = \frac{1}{v}Var(HE) + \frac{v-1}{v}Cov(HE_i, HE_j)$ the weighting $(\frac{v-1}{v})$ of the covariance is always greater than that for the individual variances (except for $v = 2$ when its equal) and this weight increases as the number of folds increases. In the studies we perform, we observe the behavior of the $E[CE]$, $Var(HE)$ (individual variances), $Cov(HE_i, HE_j)$ (covariances), $Var(CE)$ (total variance) and $E^2[CE]+Var(CE)$ (total mean squared error)[4] with varying amounts of the correlation between the input attributes and the class labels for the three classification algorithms.

Data Generation Model: The data generation model we use in the experiments is a multinomial as shown in figure 4. The details regarding the various parameters are as follows.

Number of Classes, Number of Attribute Values and Dimensionality: We fix the number of classes to 2. The number of distinct inputs n depends on the dimensionality of the input space and the number of distinct values that each attribute can take. More precisely, $n = m^d$ where d is the dimensionality and m is the number of distinct values of each attribute. The results reported are averaged over multiple dimensionalities ($d = 3$, $d = 5$, $d = 8$ and $d = 10$) with each attribute having multiple splits/values ($m = 2$ and $m = 3$) for all three algorithms (i.e. NBC, DT and KNN). Additionally, for the KNN algorithm the results are also averaged over multiple values of K (2, 5 and 8).

Input-Output Correlation: In the studies that follow the p_{ij}'s where $i \in \{1, ..., n\}$ and $j \in \{1, 2\}$ are varied and the amount of correlation between the input attributes

[4] We drop the notation which shows the space over which the moments are taken for readability purposes.

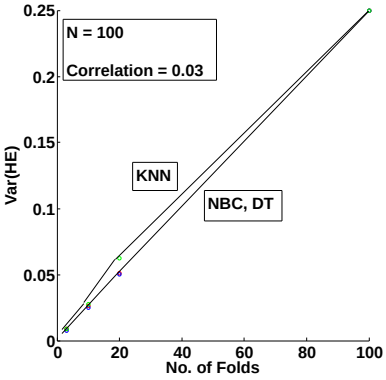

Fig. 2 Var(HE) for small sample size and low correlation.

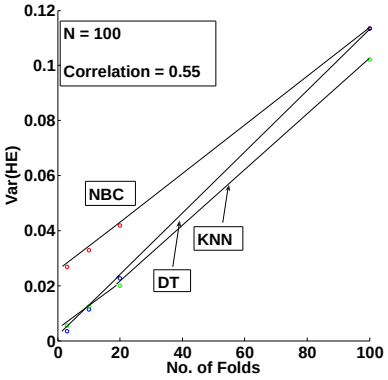

Fig. 3 Var(HE) for small sample size and medium correlation.

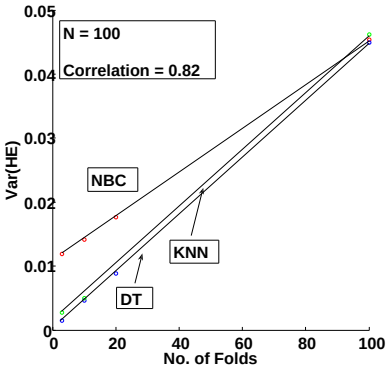

Fig. 4 Var(HE) for small sample size and high correlation.

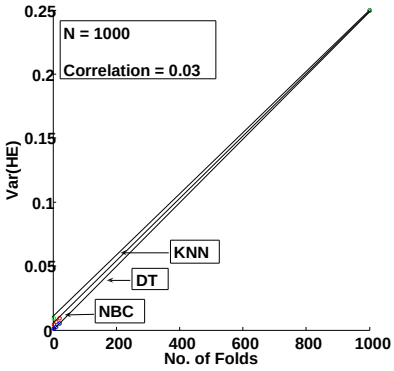

Fig. 5 Var(HE) for larger sample size and low correlation.

and the class labels is computed for each set of p_{ij}'s using the Chi-square test [4]. In particular, we define input-output correlation to be as follows: Input-output correlation $= \sum_i \frac{(p_{ij}-t_{ij})^2}{t_{ij}}$, where $t_{ij} = (\sum_{r=1}^{n} p_{rj})(\sum_{s=1}^{2} p_{is})$ is the product of the marginals for the ij^{th} entry in the table in figure 4. The exact values of the probabilities for the different correlations are given in the appendix.

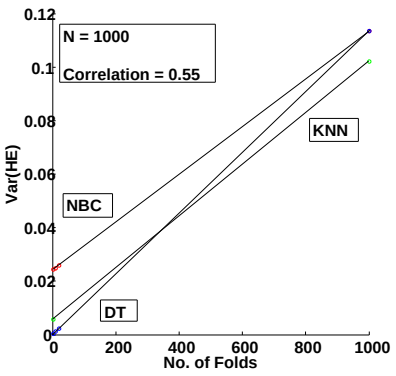

Fig. 6 Var(HE) for larger sample size and medium correlation.

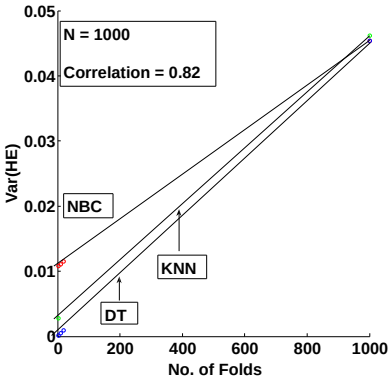

Fig. 7 Var(HE) for larger sample size and high correlation.

Dataset Size N: We initially set the dataset size to 100 and then increase it to a 1000, to observe the effect that increasing dataset size has on the behavior of these moments. The reason we choose these dataset sizes is that for the dataset sizes below and around 100 the behavior is qualitatively similar to that at 100. For larger dataset sizes say above and around 1000 the behavior is qualitatively similar to that at 1000. Secondly, the shift in trends in the behavior of cross-validation with increasing dataset size is clearly noticeable for these dataset sizes. Moreover, cross-validation is primarily used when dataset sizes are small since it has low variance and high computational cost (compared with hold-out set for example) and hence studying it under these circumstances seems sensible. The experiments reveal certain interesting facts and provide insights that assist in better understanding cross-validation.

5.1 Observations

We now explain the interesting trends that we observe through these experiments. The interesting insights are mainly linked to the behavior of the variance (in particular covariance) of CE. However, for the sake of completeness we also discuss the behavior of the expected value of CE.

5.1.1 Variance

Figures 2 to 7 are plots of the variances of the individual runs of cross-validation. Figures 8 to 13 depict the behavior of the covariance between any two runs of cross-validation. Figures 14 to 19 showcase the behavior of the total variance of cross-validation, which as we have seen is in fact a convex combination of the individual variance and the pair-wise covariance.

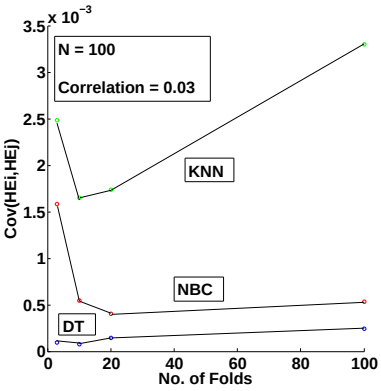

Fig. 8 $Cov(HE_i, HE_j)$ for small sample size and low correlation.

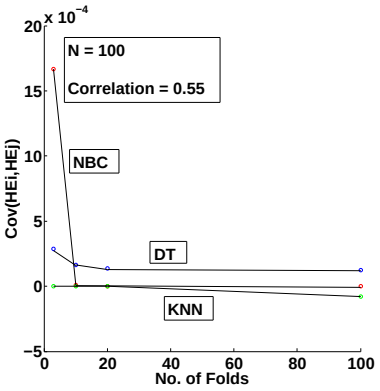

Fig. 9 $Cov(HE_i, HE_j)$ for small sample size and medium correlation.

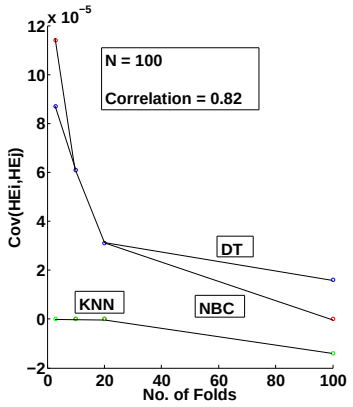

Fig. 10 $Cov(HE_i, HE_j)$ for small sample size and high correlation.

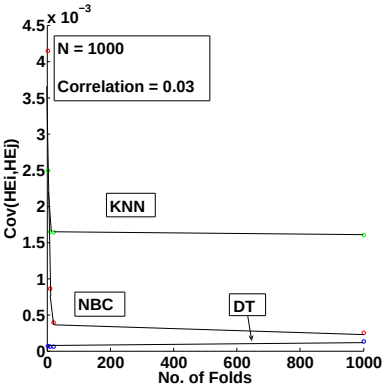

Fig. 11 $Cov(HE_i, HE_j)$ for larger sample size and low correlation.

Linear behavior of Var(HE): In Figures 2 to 7 we see that the individual variances practically increase linearly with the number of folds. This linear increase occurs since, the size of the test set decreases linearly[5] with the number of folds; and we know that CE is the average error over the v runs where the error of each run is the sum of the zero-one loss function evaluated at each test point normalized by the size of the test set. Since the test points are i.i.d. (independent and identically distributed), so are the

[5] more precisely expected test set size decreases linearly.

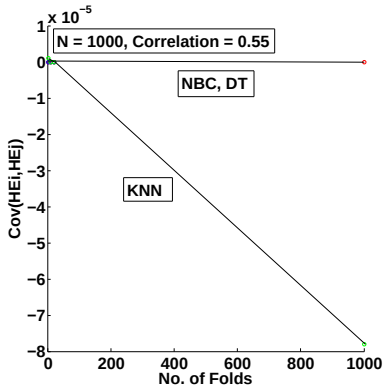

Fig. 12 $Cov(HE_i, HE_j)$ for larger sample size and medium correlation.

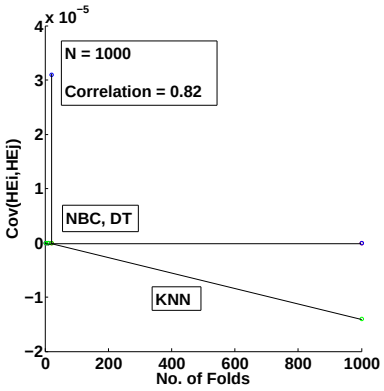

Fig. 13 $Cov(HE_i, HE_j)$ for larger sample size and high correlation.

corresponding zero-one loss functions and from theory we have that the variance of a random variable which is the sum of T i.i.d. random variables having variance $\sigma^2 < \infty$, is given by $\frac{\sigma^2}{T}$.

V-shaped behavior of $Cov(HE_i, HE_j)$: In Figure 8 we observe that the covariance first decreases as the number of folds increases from 2 to 10-20 and then increases as the number of folds approach N (called Leave-one-out (LOO) validation). This behavior has the following explanation. At low folds, for example at $v = 2$ the test set for one run is the training set for the other and vice-versa. Since each datapoint is in either one of the two partitions but cannot be in both, the partitions can be seen to have a kind of negative correlation. What we mean by this is that noticing the probability of any datapoint lying in any one of the partitions is $\frac{1}{2}$ and if P_1 and P_2 are the random variables denoting the number of datapoints in each partition then given that the total dataset size is N, the covariance between P_1 and P_2 is negative. In fact the covariance in this case is given by, $\frac{-N}{4}$. This implies that the test sets for the two runs are negatively correlated. However, the two partitions are also training sets for the two runs and hence the classifiers induced by them are also negatively correlated in terms of their prediction on individual datapoints. Since, the test sets as well as the classifiers are negatively correlated, the errors of these classifiers are positively correlated. In other words, these classifiers make roughly the same number of mistakes on the respective test sets. The reason for this is, if the two partitions are similar (i.e. say they are representative samples of the distribution) then the classifiers are similar and so are the test sets and hence both of their errors are low. When the two samples are highly dissimilar (i.e. say one is representative the other is not) the classifiers built are dissimilar and so are the test sets. Hence, the error that both of these classifiers make on their test sets which is the training set of the other classifier are high. Thus,

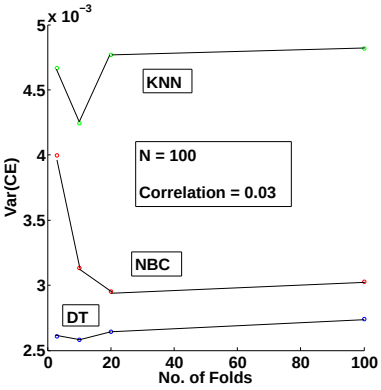

Fig. 14 Var(CE) for small sample size and low correlation.

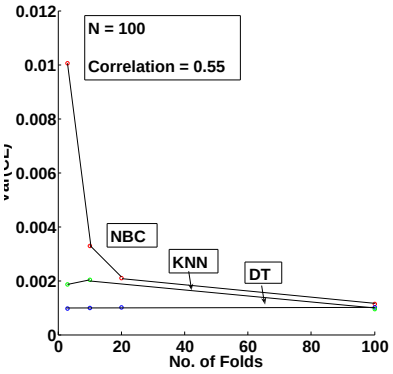

Fig. 15 Var(CE) for small sample size and medium correlation.

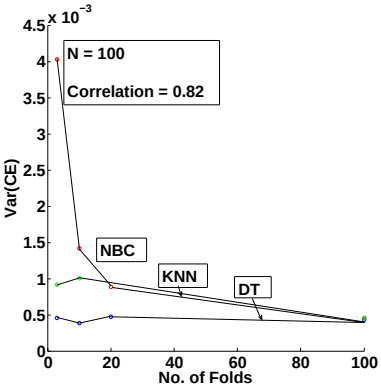

Fig. 16 Var(CE) for small sample size and high correlation.

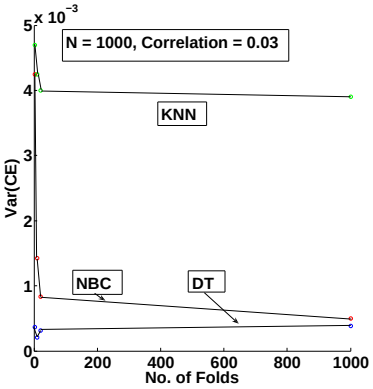

Fig. 17 Var(CE) for larger sample size and low correlation.

irrespective of the level of similarity between the two partitions the correlation between the errors of the classifiers induced by them is high.

As the number of folds increases this effect reduces as the classifiers become more and more similar due to the overlap of training sets while the test sets become increasingly unstable as they become smaller. The latter (i.e. at high folds) increase in covariance in Figure 8 is due to the case where LOO fails. If we have a majority classifier and the dataset contains an equal number of datapoints from each class, then

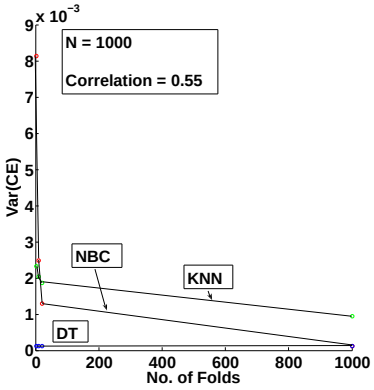

Fig. 18 Var(CE) for larger sample size and medium correlation.

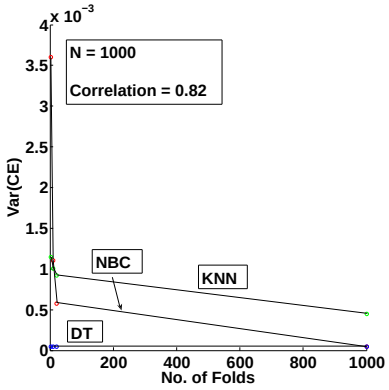

Fig. 19 Var(CE) for larger sample size and high correlation.

LOO would estimate 100% error. Since each run would produce this error the errors of any two runs are highly correlated. This effect reduces as the number of folds reduces. The classification algorithms we have chosen, classify based on majority in their final inference step i.e. locally they classify datapoints based on majority. At low input-output correlation as in Figure 8, the probability of having equal number datapoints from each class for each input is high i.e. in figure 4, the probability of $N_{i1} = N_{i2}$ $\forall i \in \{1, ..., n\}$ is high, where N_{ij} is the number of copies of the datapoint (x_i, y_j) in the multinomial. Hence, at high folds the covariance increases. Consequently, we have the V-shaped graphs as in Figure 8 which are a combination of the first effect and this second effect.

L-shaped behavior of $Cov(HE_i, HE_j)$: As we increase the correlation between the input attributes and the class labels seen in Figures 9, 10 the initial effect which raises the covariance is still dominant, but the latter effect (equal number of datapoints from each class for each input) has extremely low probability and is not significant enough to increase the covariance at high folds. As a result, the covariance drops with increasing v.

On increasing the dataset size the covariance does not increase as much (in fact reduces in some cases) in Figures 11, 12 and 13 at high folds. In Figure 11 though the correlation between the input attributes and class labels is low, the probability of having equal number datapoints from each class is low since the dataset size has increased. For a given set of parameters the probability of a particular event occurring is essentially reduced (never increased) as the number of events is increased (i.e. N increases), since the original probability mass has now to be distributed over a larger set. Hence, the covariance in Figure 11 drops as the number of folds increases. The behavior observed in Figures 12 and 13 has the same explanation as that for Figures

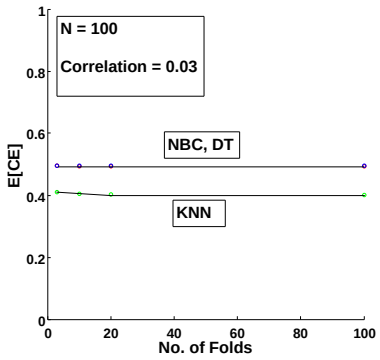

Fig. 20 E[CE] for small sample size and low correlation.

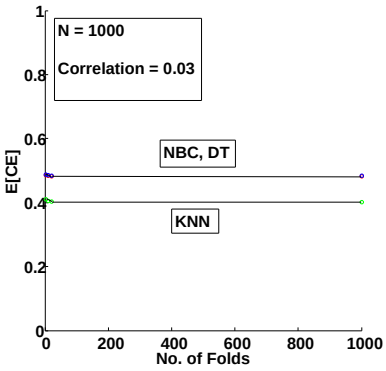

Fig. 21 E[CE] for larger sample size and low correlation.

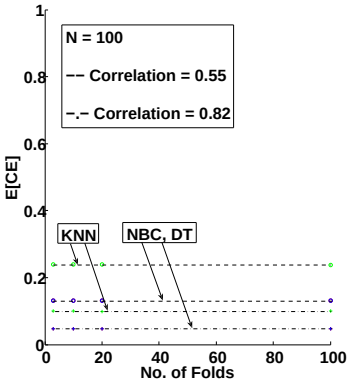

Fig. 22 E[CE] for small sample size at medium and high correlation.

9 and 10 described before.

Finally, the covariance has a V-shape for low input-output correlations and low dataset sizes where the classification algorithms classify based on majority at least at some local level. In the other cases the covariance is high initially and then reduces with increasing number of folds.

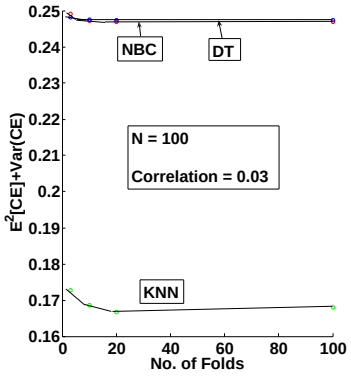

Fig. 23 $E^2[CE] + Var(CE)$ for small sample size and low correlation.

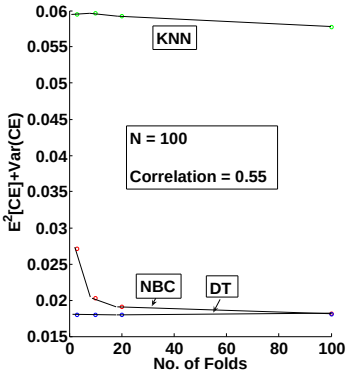

Fig. 24 $E^2[CE] + Var(CE)$ for small sample size and medium correlation.

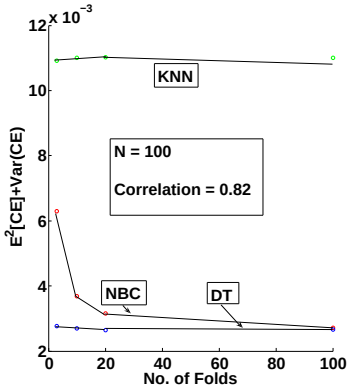

Fig. 25 $E^2[CE] + Var(CE)$ for small sample size and high correlation.

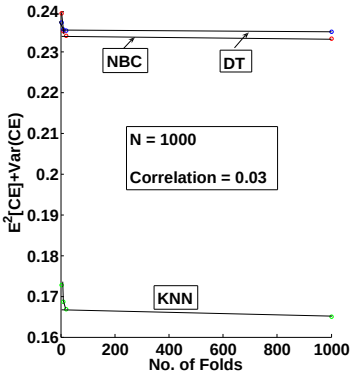

Fig. 26 $E^2[CE] + Var(CE)$ for larger sample size and low correlation.

Behavior of Var(CE) similar to covariance: Figures 14 to 19 represent the behavior of the total variance of CE. We know that total variance is given by, $Var(CE) = \frac{1}{v}Var(HE) + \frac{v-1}{v}Cov(HE_i, HE_j)$. We have also seen from Figures 2 to 7 that the $Var(HE)$'s vary almost linearly with respect to v. In other words, $\frac{1}{v}Var(HE)$ *is practically a constant with changing* v. Hence, the trends seen for the total variance are similar to those observed for the pairwise covariances with an added shift. Thus, at low correlations and low sample sizes the variance is minimum not at the extremeties but

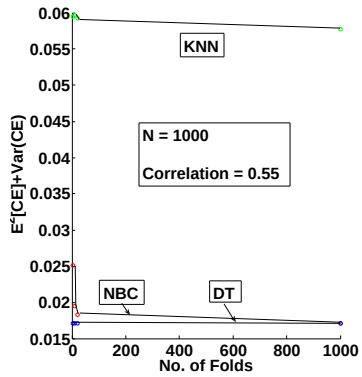

Fig. 27 $E^2[CE] + Var(CE)$ for larger sample size and medium correlation.

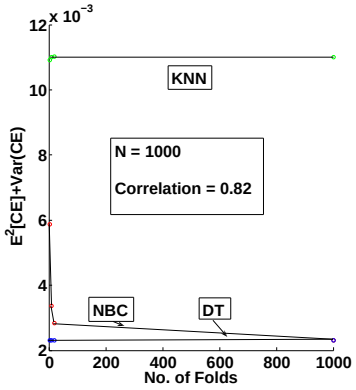

Fig. 28 $E^2[CE] + Var(CE)$ for larger sample size and high correlation.

somewhere in between, say around 10-20 folds as seen in 14. In other cases, the total variance reduces with increasing number of folds.

5.1.2 Expected Value

Figures 20 to 22 depict the behavior of the expected value of CE for different amounts of correlation between the input attributes and the class labels and for two different sample sizes. The behavior of the expected value at medium and high correlations for small and large sample sizes is the same and hence we plot these scenarios only for small sample sizes as shown in Figure 22. From the figures we observe that as the correlation increases the expected CE reduces. This occurs since the input attributes become increasingly indicative of a particular class. As the number of folds increase the expected value reduces since the training set sizes increase on expectation, enhancing classifier performance.

5.1.3 Expected value square + Variance

Figures 23 to 28 depict the behavior of CE. In Figure 23 we observe that the best performance of cross-validation is around 10-20 folds. In the other cases, the behavior improves as the number of folds increases. In Figure 23 the variance at high folds is large and hence the above sum is large for high folds. As a result we have a V-shaped curve. In the other Figures the variance is low at high folds and so is the expected value and hence the performance improves as the number of folds increases.

6 Conclusion

In summary, we observed the behavior of cross-validation under varying amounts of input-output correlation, varying sample size and with varying number of folds. We observed that at low correlations and for low sample sizes (a characteristic of many real life datasets) 10-20 fold cross-validation was the best while for the other cases increasing the number of folds helped enhance performance. Additionally, we provided in depth explanations for the observed behavior and commented that the explanations for the behavior of covariance were especially relevant to classification algorithms that classify based on majority at a global level (e.g. majority classifiers) or at least at some local level (e.g. DT classification at the leaves). The other interesting fact was that all the experiments and the insights were a consequence of the theoretical formulas that were derived previously. We hope that non-asymptotic studies like the one presented will assist in better understanding popular prevalent techniques, in this case cross-validation.

References

1. Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross validation. *Journal of Machine Learning Research*, 2003.
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. L. Breiman. Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, 1996.
4. J. Connor-Linton. Chi square tutorial. http://www.georgetown.edu/faculty/ballc/webtools/web_chi_tut.html, 2003.
5. L. Devroye, L. Gyorfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.
6. A. Dhurandhar and A. Dobra. Probabilistic characterization of nearest neighbor classifier. Technical Report at www.cise.ufl.edu, 2007.
7. A. Dhurandhar and A. Dobra. Probabilistic characterization of random decision trees. *Journal of Machine Learning Research*, 9, 2008.
8. 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.
9. B. Efron. How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81:461–470, 1986.
10. B. Efron. The estimation of prediction error: Covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99:619–642, 2004.
11. A. Elisseeff and M. Pontil. *Learning Theory and Practice*, chapter Leave-one-out error and stability of learning algorithms with applications. IOS Press, 2003.
12. C. Goutte. Note on free lunches and cross-validation. *Neural Computation*, 9(6):1245–1249, 1997.
13. I. Guyon. Nips. Discussion: Open Problems, 2002.
14. M. Kearns and D. Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. In *Computational Learing Theory*, 1997.
15. R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *In Proceedings of the Fourteenth IJCAI*, 1995.
16. M. Markatou, H. Tian, S. Biswas, and G. Hripcsak. Analysis of variance of cross-validation estimators of the generalization error. *J. Mach. Learn. Res.*, 6:1127–1168, 2005.
17. A. Moore and M. Lee. Efficient algorithms for minimizing cross validation error. In *International Conference on Machine Learning*, pages 190–198, 1994.
18. M. Plutowski. Survey: Cross-validation in theory and in practice. www.emotivate.com/CvSurvey.doc, 1996.
19. J. Schneider. Cross validation. http://www.cs.cmu.edu/ schneide/tut5/node42.html, 1997.
20. J. Shao. Linear model selection by cross validation. *JASA*, 88, 1993.

21. M. Stone. Asymptotics for and against cross-validation. *Biometrika*, 64:29–35, 1977.
22. V. Vapnik. *Statistical Learning Theory*. Wiley & Sons, 1998.
23. H. Zhu and R. Rohwer. No free lunch for cross validation. *Neural Computation*, 8(7):1421–1426, 1996.

Acknowledgements

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

Appendix A.

The moments of CE are related to the moments of GE and hence if the moments of GE are known moments of CE can be computed. The moments of GE are given by,

$$E_{D(N)}[GE(\zeta)] = \sum_{x \in \mathcal{X}} P[x] \sum_{y \in \mathcal{Y}} P_{D(N)}[\zeta(x) = y] \, P[Y(x) \neq y]$$

$$E_{D(N)\times D(N)}[GE(\zeta)GE(\zeta')] = \sum_{x \in \mathcal{X}} \sum_{x' \in \mathcal{X}} P[x] \, P[x'] \cdot \sum_{y \in \mathcal{Y}} \sum_{y' \in \mathcal{Y}} P_{D(N)\times D(N)}[\zeta(x) = y \wedge \zeta'(x') = y'] \cdot P[Y(x) \neq y] \, P[Y(x') \neq y']$$

The terms $P_{D(N)}[\zeta(x) = y]$ and $P_{D(N)\times D(N)}[\zeta(x) = y \wedge \zeta'(x') = y']$ in the expressions for the first moment and second moment respectively, are the only terms that are dependent on both the classification algorithm and the data distribution (i.e. underlying distribution and modeling process). The other terms are dependent only on the data distribution and hence remain unchanged for different classification algorithms. Moreover, they are straightforward to compute. Consequently, for each of the 3 algorithms namely, NBC, DT and KNN we need to provide customized expressions only for the stated two terms.

.1 DT

We find the moments for decision trees built using random attribute selection and with purity as stopping criteria.

We consider the dimensionality of the input space to be d. $A_1, A_2, ..., A_d$ are the corresponding discrete attributes or continuous attributes with predetermined split points. $a_1, a_2, ..., a_d$ are the number of attribute values/the number of splits of the attributes $A_1, A_2, ..., A_d$ respectively. m_{ij} is the i^{th} attribute value/split of the j^{th} attribute, where $i \leq a_j$ and $j \leq d$. Let $y_1, y_2, ..., y_k$ be the class labels representing k classes and N the sample size.

Purity: This stopping criteria implies that we stop growing the tree from a particular split of a particular attribute if all datapoints lying in that split belong to the same class. We call such a path pure else we call it impure. In this scenario, we could have paths of length 1 to d depending on when we encounter purity (assuming all datapoints don't lie in 1 class). Thus, we have the following two separate checks for paths of length d and less than d respectively.

a) Path $m_{i1}m_{j2}...m_{ld}$ present iff the path $m_{i1}m_{j2}...m_{l(d-1)}$ is impure and attributes $A_1, A_2, ..., A_{d-1}$ are chosen above A_d, or $m_{i1}m_{j2}...m_{s(d-2)}m_{ld}$ is impure and attributes $A_1, A_2, ..., A_{d-2}, A_d$ are chosen above A_{d-1}, or ... or $m_{j2}...m_{ld}$ is impure and attributes $A_2, ..., A_d$ are chosen above A_1.

This means that if a certain set of $d-1$ attributes are present in a path in the tree then we split on the d^{th} attribute iff the current path is not pure, finally resulting in a path of length d.

b) Path $m_{i1}m_{j2}...m_{lh}$ present where $h < d$ iff the path $m_{i1}m_{j2}...m_{lh}$ is pure and attributes $A_1,\ A_2,...,\ A_{h-1}$ are chosen above A_h and $m_{i1}m_{j2}...m_{l(h-1)}$ is impure or the path $m_{i1}m_{j2}...m_{lh}$ is pure and attributes $A_1,\ A_2,...,\ A_{h-2},\ A_h$ are chosen above A_{h-1} and $m_{i1}m_{j2}...m_{l(h-2)}m_{lh}$ is impure or ... or the path $m_{j2}...m_{lh}$ is pure and attributes $A_2,...,\ A_h$ are chosen above A_1 and $m_{j2}...m_{lh}$ is impure.
This means that if a certain set of $h-1$ attributes are present in a path in the tree then we split on some h^{th} attribute iff the current path is not pure and the resulting path is pure.

The above conditions suffice for "path present" since the purity property is anti-monotone and the impurity property is monotone.

Some of the conditions above are sample dependent while others are sample independent or in other words depend on the attribute selection method. Let *s.c.c.s.* be an abbreviation for stopping criteria conditions that are sample dependent. With this we have the following expression for the probability used in the first moment,

$$\begin{aligned}P[\zeta(x)=y_i] &= \sum_p P[ct(path_p y_i) > ct(path_p y_j), path_p exists,\ \forall j \neq i,\ i,j \in [1,...,k]] \\ &= \sum_p \frac{P[ct(path_p y_i) > ct(path_p y_j), s.c.c.s.,\ \forall j \neq i,\ i,j \in [1,...,k]]}{dC_{h_p-1}(d-h_p+1)}\end{aligned}$$

where h_p is the length of the path indexed by p and ct(.) is the number of datapoints with the specified attribute values. The joint probability of comparing counts and *s.c.c.s.* can be computed from the underlying joint distribution. The probability for the second moment when the trees are different is given by,

$$\begin{aligned}&P\left[\zeta(x)=y_i \wedge \zeta'(x')=y_v\right] \\ &= \sum_{p,q} P[ct(path_p y_i) > ct(path_p y_j), path_p exists, ct(path_q y_v) > ct(path_q y_w), path_q exists, \\ &\quad \forall j \neq i,\ \forall w \neq v,\ i,j,v,w \in [1,...,k]] \\ &= \sum_{p,q} \frac{P[ct(path_p y_i) > ct(path_p y_j), ct(path_q y_v) > ct(path_q y_w), s.c.c.s., \forall j \neq i,\ \forall w \neq v,\ i,j,v,w \in [1,...,k]]}{dC_{h_p-1}dC_{h_q-1}(d-h_p+1)(d-h_q+1)}\end{aligned}$$

where h_p and h_q are the lengths of the paths indexed by p and q.

.2 NBC

Consider $x1,\ x2,...,\ xd$ are d explanatory attributes and y_1 and y_2 are the class labels. N_1 and N_2 denote the number of datapoints in class y_1 and y_2 respectively in the training set. N_{ij}^{xk} denotes number of copies of the input output pair (xk_i, y_j) where xk_i is the i^{th} value of attribute xk, $j \in \{1,2\}$ and $k \in \{1,2,...,d\}$. With this the 2 probabilities required for the computation of the first 2 moments are given by,

$$\begin{aligned}&P[\zeta(x)=y_1] \\ &= P[N_2^{(d-1)} N_{i_1 1}^{x1} N_{i_2 1}^{x2} ... N_{i_d 1}^{xd} > N_1^{(d-1)} N_{i_1 2}^{x1} N_{i_2 2}^{x2} ... N_{i_d 2}^{xd}]\end{aligned}$$

$$\begin{aligned}&P\left[\zeta(x)=y_1 \wedge \zeta'(x')=y_2\right] \\ &= P[N_2^{(d-1)} N_{i_1 1}^{x1} N_{i_2 1}^{x2} ... N_{i_d 1}^{xd} > N_1^{(d-1)} N_{i_1 2}^{x1} N_{i_2 2}^{x2} ... N_{i_d 2}^{xd}, \\ &\quad N_2^{(d-1)} N_{j_1 1}^{x1} N_{j_2 1}^{x2} ... N_{j_d 1}^{xd} \leq N_1^{(d-1)} N_{j_1 2}^{x1} N_{j_2 2}^{x2} ... N_{j_d 2}^{xd}]\end{aligned}$$

where x is the input vector $(i_1, i_2, ..., i_d)$ and x' is the input vector $(j_1, j_2, ..., j_d)$. It is easy to see that the above formulas can be used for any valid input with the insertion of appropriate indices.

.3 KNN

The scenario wherein x is classified into class y_j given $j \in \{1, 2, ..., v\}$ depends on two factors; 1) the kNN's of x 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. The second factor is always determined by the sample. The $P[\zeta(x)=y_j]$ is the probability of all possible ways that input x can be classified into class y_j, given the joint distribution over the input-output space. This probability for x 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 y_j, over all possible kNN's that the input can have. Formally,

$$P[\zeta(x)=y_j] = \sum_{q \in Q} P[q, c(q,j) > c(q,t), \forall t \in \{1, 2, ..., v\}, t \neq j]$$

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 y_b. The $P[\zeta(x)=y \wedge \zeta'(x')=y']$ 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,

$$P\left[\zeta(x)=y_j \wedge \zeta'(x')=y_w\right] = \sum_{q \in Q} \sum_{r \in R} P[q, c(q,j) > c(q,t), r, c(r,w) > c(r,s) \quad \forall s, t \in \{1, 2, ..., v\}, t \neq j, s \neq w]$$

where q and r are sets of kNN's of x and x' respectively. Q and R are sets containing all possible q and r respectively. $c(.,.)$ has the same connotation as before. The complexity of the above formulas is reduced to low degree polynomials for sample independent distance metrics ([6]).

.4 Setting Multinomial Probabilities

In our experiments the input-output correlation was varied from low to medium to high where low meant the correlation was 0.03, medium meant the correlation was 0.55 and high meant the correlation was 0.82. These correlations were obtained (using the chi-square formula) by setting the multinomial probabilities as follows: Given n distinct inputs and 2 classes as in figure 4, for high correlation p_{i1} $\forall i \in \{1, ..., \frac{n}{2}\}$ and p_{i2} $\forall i \in \{\frac{n}{2}, ..., n\}$ were set to $\frac{1}{1.05n}$. The remaining probabilities were all equal with the constraint that all the probabilities must sum to 1. For medium correlation p_{i1} $\forall i \in \{1, ..., \frac{n}{2}\}$ and p_{i2} $\forall i \in \{\frac{n}{2}, ..., n\}$ were set to $\frac{1}{1.15n}$. The remaining probabilities were all equal with the constraint that all the probabilities must sum to 1. Similarly, for low correlation p_{i1} $\forall i \in \{1, ..., \frac{n}{2}\}$ and p_{i2} $\forall i \in \{\frac{n}{2}, ..., n\}$ were set to $\frac{1}{1.9n}$. The remaining probabilities were again all equal with the constraint that all the probabilities must sum to 1.