Chapter 3

The Feature Selection Problem

Feature selection should be treated as an integral part of dataset selection. It was pointed out in the last chapter that the use of a good set of predictive features leads to the reduction of the variance component of the prediction error. This chapter provides an overview of the feature selection problem for classification tasks in predictive data mining. A review of the available methods for feature selection from small datasets is provided. The methods discussed fall into two categories. The methods in the first category have been studied by researchers in the context of small datasets. The methods in the second category have been studied to specifically address feature selection for data mining for high dimensional datasets and for large datasets. The dangers of using a single sample to determine relevant features are highlighted. An analysis is conducted of commonly used methods of measuring class-feature correlations and more robust measures of class-feature correlations are discussed. Existing methods for validation of correlation values are also discussed.

This chapter is organised as follows: The need for feature selection is discussed in section 3.1. Methods for implicit and explicit feature selection are respectively discussed in sections 3.2 and 3.3. Merit measures for heuristic feature subset search are given in section 3.4. Sections 3.5 and 3.6 respectively provide a discussion of correlation measurement and validation methods for correlations. Section 3.7 concludes the chapter.

3.1 The need for feature selection

The problem of feature subset selection is concerned with finding a subset of the original features of a dataset, such that an induction algorithm running on data containing only the selected features will generate a predictive model that has the highest possible accuracy. It is essential to select a subset of those features which are most relevant to the prediction problem and are not redundant (Hand et al, 2001; Hall, 1999, 2000; Liu & Motoda, 1998; Blum & Langley, 1997; Aha & Bankert, 1996).

Feature selection is central to training dataset selection since one of the motivating factors for training dataset selection is to improve predictive accuracy through variance reduction. This section provides a discussion of the need for feature selection as well as definitions of relevance and redundancy as reported in the literature on feature selection for machine learning and data mining. Section 3.1.1 provides definitions of feature relevance and redundancy. Section 3.1.2 discusses a major problem for predictive modeling called the curse of dimensionality.

3.1.1 Feature relevance and redundancy

It is generally agreed that a predictive model should be constructed using a subset of features which are most relevant to the prediction problem and are not redundant (Hand et al, 2001; Hall, 1999, 2000; Liu & Motoda, 1998; Blum & Langley, 1997; Aha & Bankert, 1996). Blum and Langley (1997) have provided definitions of *relevance*, *strong relevance* and *weak relevance*. A feature f_i is *relevant* if a change in the feature's value can result in a change in the value of the predicted (class) variable. A feature f_i is *strongly relevant* if the use of f_i in the predictive model eliminates the ambiguity in the classification of instances. A feature f_i is *weakly relevant* if f_i becomes *strongly relevant* when a subset of the features is removed from the set of available features. By implication, a feature is *irrelevant* if it is not *strongly relevant* and it is not *weakly relevant*. Koller and Sahami (1996) have provided a definition of *redundancy* for a feature. A feature f_i is *redundant* relative to the class variable *C* and a second feature f_j if f_i has stronger predictive power for f_j than for the class variable *C*. Koller and Sahami (1996) have used the term *Markov blanket* to refer to the above relationship between the features f_j and f_i, that is f_j is a *Markov blanket* for f_i.

For purposes of making feature selection decisions however, many researchers (e.g. Ooi et al, 2007; Yu & Liu, 2004; Blum & Langley, 1997; Hall, 1999,2000) have interpreted a *relevant* feature as one which is highly correlated with the class variable. Ooi et al (2007) and Hall (1999, 2000) have interpreted a redundant feature as one which is highly correlated with all the other features. Yu and Liu (2004) and, Koller and Sahami (1996) have however implemented the above definition of *redundancy* (based on the *Markov blanket* property) in feature selection.

3.1.2 The curse of dimensionality

Hand et al (2001) have discussed the problem of the *curse of dimensionality*. This problem is defined as the exponential rate of growth of the number of unit cells in the instance space as the number of predictive features increases. The curse of dimensionality reduces the density of instances in the instance space. Recall from section 2.2.1 that the instance space is the *d*-dimensional space defined by the *d* predictor variables. Reduction of the density of instances in the instance space causes instances to appear to be very far away from each other. This makes it difficult for discriminative classification algorithms to establish decision regions and decision boundaries for classes. It also becomes more difficult for probabilistic classification models to estimate probability densities in the different regions of the instance space.

The reduction of the number of features reduces the size of the instance space, and therefore also decreases the complexity of the prediction problem. Secondly, according to the PAC theory (Mitchell, 1997), as the hypothesis space size decreases in size, so does the sample complexity.

3.2 Implicit feature selection

Decision tree algorithms have the capability to implicitly identify the most predictive features as the tree is constructed. In addition, a decision tree is normally pruned so that it retains only those features which provide statistically significant predictive power (Osei-Bryson, 2004, 2007; Breiman et al, 1984). Kohavi and John (1997) have reported feature selection studies which have revealed that decision tree algorithms are not always able to eliminate irrelevant features. The studies reported by Kohavi and John (1997) on credit approval and diabetes datasets from the UCI Machine Learning repository (Ascuncion & Newman, 2007; Blake & Merz, 1998) have shown that the performance of decision trees constructed by the C4.5 algorithm deteriorates significantly when a single irrelevant feature is added to the dataset. Langley (1994) has observed that artificial neural networks (ANNs) and the Naïve Bayes (NB) algorithms also perform an implicit ranking as they build classifiers. ANNs and NB assign larger weights to the more relevant features and smaller weights to the less relevant ones. A very important point to emphasize here is that, even though many inductive algorithms perform implicit feature selection, all induction algorithms do benefit from explicit feature selection, before the algorithm is presented with the data.

3.3 Explicit feature selection

Explicit feature selection involves the use of a separate step to select those features that are considered relevant for a predictive modeling task. Specific to data mining there may be hundreds or thousands of features in a dataset. All potentially relevant features must be identified first. The identification is typically done by a domain expert (Guyon & Elisseef, 2003). The task of the domain expert is to select those variables that are known to have a bearing on the domain for the prediction task. As an example, van der Putten and van Someren (2004) have quoted the winner of the COIL 2000 competition who stated that only the variables representing wealth and personal behaviour of individuals were useful for the competition dataset. After the initial selection of features, a second step is conducted so that the most effective (predictive) features are selected from the pool of potentially relevant features (Guyon & Elisseef, 2003). Section 3.3.1 presents the categories of feature selection methods. Wrapper methods are discussed in section 3.3.2. Methods that use pure ranking are presented in section 3.3.3. Heuristics search methods and relevance and redundancy analysis methods are respectively discussed in sections 3.3.4 and 3.3.5. Feature selection methods for large datasets are discussed in section 3.3.6.

3.3.1 Categories of feature selection methods

Feature selection methods may be categorized as wrapper or filtering methods (Hall, 1999; Kohavi & John, 1997). Wrapper methods incorporate model construction with feature selection, and select that subset of features which provides a model with the highest predictive performance (Blum & Langley, 1997; Kohavi & John, 1997). Filtering methods on the other hand, select feature subsets without constructing predictive models from these features (Ooi et al, 2007; Yu & Liu, 2004; Guyon & Eliseeff, 2003; Hall, 1999; Blum & Langley, 1997). Three filtering methods are discussed in this section. The first method called *pure ranking*, involves sorting the list of features in descending order of relevance and then selecting the top *w* features or selecting features whose level of relevance is above a user specified threshold (Yu & Liu, 2004). The second method, called *feature subset search*, involves a forward search or backward search based on a list of ranked features in order to determine the best subset of features which maximises relevance and minimises redundancy (Ooi et al, 2007; Hall, 1999, 2000; Blum & Langley, 1997). The third filtering method involves *relevance* and *redundancy analysis* as two distinct steps (Yu & Liu, 2004). The rest of this section provides a discussion of wrapper and filtering methods for feature selection.

3.3.2 Feature selection using wrapper methods

Wrapper methods incorporate model construction with feature selection (Blum & Langley, 1997; Kohavi & John, 1997). For wrapper methods, different feature subsets are selected, a predictive model is constructed for each feature subset and the feature subset which produces the model with the highest predictive performance is selected. The accuracy for different feature subsets is measured using 10-fold cross validation (Blum & Langley, 1997). Wrapper methods have typically been used for small datasets with a small number of features. It has been argued that wrapper methods are not suitable for large datasets as encountered in data mining (Hall, 1999) or datasets of high dimensionality (Yu & Liu, 2004) due to the intensive computational requirements. Even though the research reported in this thesis focussed on filtering methods, it is the author's opinion that when many samples are used for model construction and testing for the wrapper approach then more reliable feature selection should be achieved.

3.3.3 Feature selection based on pure ranking

Feature ranking involves two steps. In the first step, a value is assigned to each feature to indicate its level of relevance to the prediction task. In the second step, the list of features is sorted and the top *w* features are selected. A commonly used measure of relevance is the correlation of the feature to the class. To compute the correlation between a numeric-valued feature and the class variable, Pearson's correlation coefficient is commonly used (Ooi et al, 2007; Hall, 1999, 2000). To compute the correlation between a qualitative feature and the class variable, the symmetrical uncertainty coefficient may be used (Yu & Liu, 2004; Hall, 1999, 2000). Guyon and Elisseeff (2003) and Bekkerman et al (2003) have observed that various feature selection algorithms include feature ranking as a preliminary step. The purpose of this preliminary step is to identify those features that have the potential to appear in the final subset of selected features. Various feature selection methods, on the other hand, simply use feature ranking as the selection method (Guyon & Elisseeff, 2003).

3.3.4 Feature selection based on heuristic search

Heuristic search (Luger & Stubblefield, 1993; Pearl, 1984) is the process of intelligently narrowing the search through a potentially very large search space of solutions in order to identify a satisfactory solution. At every decision point in the search, a heuristic search procedure employs a merit (heuristic) measure to determine the best path to expand in the search space. For the problem of feature subset search the space of all possible problems is the set of all possible combinations (the power set) of the features in the set of candidate features. The candidate features are those features that have been pre-selected through a process of ranking as discussed in section 3.3.1. Each state in the search space specifies a possible subset of features (Blum & Langley, 1997).

Algorithms for feature subset search are classified as forward selection or backward selection algorithms. The initial state for forward selection algorithms is one where no feature has been selected. For backward elimination algorithms on the other hand, the initial state is one where all features are selected. A hill-climbing search is commonly conducted. The state which currently maximises the measure of merit is selected for further expansion. Commonly used measures include information gain and merit measures based on the class-feature and feature-feature correlations (Ooi et al, 2007; Hall, 1999, 2000). For forward search, stopping criteria include stopping when addition of a new feature does not result in any significant increase in the employed measure (Hall, 1999, 2000), or when a pre-specified number of features has been selected (Ooi et al, 2007).

3.3.5 Feature selection using relevance and redundancy analysis

Yu and Liu (2004) have proposed the method of *relevance* and *redundancy analysis* for feature selection aimed at datasets of very high dimensionality. Yu and Liu (2004) have argued that heuristic subset search and wrapper methods are not feasible for high dimensional datasets due to the quadratic time complexity of heuristic search algorithms. The feature selection method proposed by Yu and Liu (2004) consists of two distinct steps, namely *relevance analysis* and *redundancy analysis*. For relevance analysis class-feature correlations are used as a basis to eliminate all features whose level of correlation to the class variable is below a user-specified threshold. The features selected in the *relevance analysis* step are used as input to the *redundancy analysis* step. *Redundancy analysis* aims to select those features that are relevant with respect to the class variable and are not redundant with respect to any other relevant feature. For each relevant feature f_i, every feature f_j which has a smaller class-feature correlation than f_i (lower relevance than f_i), but has a high feature-feature correlation with f_i (more strongly correlated to f_i than to the class variable) is eliminated.

Yu and Liu (2004) have demonstrated that even though this method has quadratic time complexity in the worst case, in practice the time complexity is close to linear time when many redundant features are present. The studies reported in this thesis were limited to datasets of moderately high dimensionality. It will be useful in future to study how the validation methods proposed in this thesis can be adapted to feature selection for very high dimensional datasets.

3.3.6 Feature selection for large datasets

For feature selection for small datasets all the instances in the dataset are used in the selection process. Feature selection from large datasets poses new challenges for feature selection. A dataset may be large because it consists of a large number of instances, or a large number of potentially predictive features, or both. From a computational perspective, the time complexity of feature selection algorithms makes it infeasible to use all of the data in a large dataset (Liu & Setiono, 1998a, 1998b). From a statistical perspective, the problems of massive search (Smyth, 2001) which were discussed in chapter 2 make it infeasible to use all of the data. Liu and Setiono

(1998a, 1998b) have proposed a stochastic (probabilistic) method of feature selection for large datasets of high dimensionality. The method employs random feature subset generation and evaluation in conjunction with dataset sampling. At each step of the process a sample of dataset instances is created and a subset of randomly selected features is generated and evaluated. The process stops when the selected feature subset is established to be optimal.

3.4 Merit measures for heuristic search of feature subsets

Heuristic search for feature selection was discussed in section 3.3.2. Suppose that at the current step of heuristic search, *w-1* features $f_1 .., f_{w-1}$ have been selected from the candidate set of features W and $u = |W| - w - 1$ features are still unselected. In order to select the next feature f_w, feature subsets $FS_i = \{f_1 .., f_{w-1}, f_w\}, i = 1.u$ are created so that for each subset FS_i the feature f_w is one of the u features that are still unselected. A mathematical function is typically used to compute a measure of merit which guides the heuristic search in the selection of the best feature subset FS^*. The correlation-based feature selection (CFS) method proposed by Hall (1999, 2000) uses the merit measure defined as

$$Merit_{CFS} = \frac{w.\overline{corr_{cf}}}{\sqrt{w + w(w-1)\,\overline{corr_{ff}}}} \qquad (3.1)$$

where $\overline{corr_{cf}}$ is the mean correlation between each feature and the class variable, $\overline{corr_{ff}}$ is the mean correlation between the features in subset *FS* and, *w* is the number of features in the subset *FS*. The numerator on the right hand side of equation (3.1) measures the level of relevance of the feature subset, while the denominator measures the level of redundancy of the feature subset. The differential Prioritisation (DP) method, proposed by Ooi et al (2007) uses the merit measure defined as

$$Merit_{DP} = (\overline{corr_{cf}})^{\alpha}.(RD)^{1-\alpha} \qquad (3.2)$$

where

$$RD = \frac{1}{w^2} \sum_{fi,fj \in F, fi \neq fj} (1 - | corr_{fifj} |) \tag{3.3}$$

and $| corr_{fifj} |$ is the magnitude of the correlation between two features. The first term on the right hand side of equation (3.2) measures the level of relevance, while the second term measures the level of redundancy of the feature subset. The parameter α is used to control the levels feature relevance and redundancy based on the user's preference. The merit measures of equations (3.1) and (3.2) both reflect the fact that the subset of selected features should have a high level of relevance and a low level of redundancy. The main difference between the two equations is that the relative importance of relevance and non-redundancy are fixed in equation (3.1) while equation (3.2) allows the analyst to specify the relative importance of relevance and non-redundancy through the parameter α.

The correlation coefficients $corr_{cf}$ and $corr_{ff}$ are computed using either Pearson's correlation coefficient for quantitative features or the symmetrical uncertainty coefficient for qualitative features. For two quantitative features *X* and *Y*, the correlation is measured using Pearson's correlation coefficient, which is defined as

$$r_{XY} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{(n-1) S_X S_Y} \tag{3.4}$$

where $\bar{x}$ and $\bar{y}$ are the sample means for *X* and *Y* respectively, S_X and S_Y are the sample standard deviations for *X* and *Y*, and *n* is the size of the sample used to compute the correlation coefficient.

In general, the level of association between two qualitative variables *X* and *Y* can be established using measures derived from Pearson's χ^2 statistic, such as the ϕ statistic and Cramer's *V* statistic (Giudici, 2003). These measures of association have a similar interpretation as a correlation coefficient for quantitative features (Giudici, 2003). The symmetrical uncertainty (*SU)* coefficient derived from the entropy function is an alternative measure of association between two qualitative features and also has a similar interpretation as a correlation coefficient for quantitative variables (Yu & Liu, 2004; Hall, 1999, 2000). The symmetrical uncertainty coefficient *SU* given in equation (3.5) is defined in terms of *E(X)* and *E(Y)* (the entropy in *X* and Y respectively) and *E(X|Y)* (the entropy of *X* conditioned on Y). The definitions for *E(X), E(X|Y)* and other related definitions of entropy are given in appendix B. The *SU* coefficient is defined as

$$SU = 2.0x\left[\frac{E(X) - E(X \mid Y)}{E(X) + E(Y)}\right] \tag{3.5}$$

The details for the computation of Pearson's χ^2 statistic, the ϕ statistic and Cramer's *V* statistic are given in appendix B. The *SU* coefficient was used in the experiments for this thesis as it is more commonly used in feature selection studies (Yu & Liu, 2004; Hall, 1999, 2000).

When one feature *X* is qualitative and the other feature *Y* is quantitative, a weighted Pearson's correlation is used. If the qualitative feature *X* has *V* levels, $L_1 \ldots L_V$, then *V* binary features $B_1 \ldots B_V$ are created through a process called binarisation. Each of the binary features is then correlated with the quantitative feature *Y*. The binary feature B_i is assigned the value *1* when *X* has level L_i and *0* otherwise. The weighted correlation coefficient between *X* and *Y* is computed as

$$corr_{XY} = \sum_{i=1}^{V} P_r(X = L_i).corr_{Bi,Y} \tag{3.6}$$

where $P_r(X = L_i)$ is the prior probability that *X* has the level L_i and $corr_{Bi,Y}$ is the correlation coefficient between a binary feature and the variable *Y*.

The computation of correlation coefficients using equations (3.6) is feasible for qualitative features with few distinct levels. If a qualitative feature has many levels (e.g. 20 and above) then the number of binary features to be created becomes excessively large, which in turn increases the computational time for the correlation coefficients. Equation (3.6) is especially useful for computing correlations between quantitative features and the class variable. Since many classification tasks have few classes the computations for equation (3.6) do not pose a problem.

3.5 Measuring correlations

In general, there are three common methods of measuring the correlation between two quantitative random variables *X* and *Y*: Pearson's correlation coefficient, Spearman's *rho,* and Kendall's *tau* (Wilcox, 2001). Each of these correlation measures is exactly zero when *X* and *Y* are independent, and have values that range between *-1* and *+1* to indicate the level and direction of the correlation. Pearson's correlation coefficient is commonly used to estimate the magnitude of the association between the features and the class for a dataset (Ooi et al, 2007; Hall, 1999, 2000). The main advantage of Pearson's correlation is that it is very efficient to compute, compared to the other correlation measures. However, for many datasets used in data mining the meaning of Pearson's correlation coefficient may not be what one expects. The problems associated with Pearson's correlation coefficient and the advantages of using robust measures of correlation are discussed in this section. The problems associated with Pearson's correlation coefficient and robust measures of correlation are respectively presented in sections 3.5.1 and 3.5.2.

3.5.1 Problems with Pearson's correlation coefficient

Pearson's product moment correlation coefficient for a data sample is computed as shown in equation (3.4). The computation involves the sample mean, sample variance and sample covariance. Furthermore, the sample mean, variances and covariances can be computed in a single pass of the dataset. Wilcox (2001) has observed that Pearson's correlation coefficient is the best estimator of the correlation between the random variables *X* and *Y* when *X* and *Y* have normal probability distributions. Wilcox (2001) has defined the *finite sample breakdown point* of a statistic computed from a sample as the smallest proportion of outliers in the sample required to make the value of the statistic arbitrarily large or arbitrarily small. Wilcox (2001) has demonstrated that the *finite sample breakdown point* for the sample mean and sample variance is *1/n*, where *n* is the sample size. This means that a single outlier can cause these measures to be arbitrarily large or small. For Pearson's correlation coefficient, Wilcox (2001) has also demonstrated that a single outlier can mask an otherwise strong association between *X* and *Y*.

The above observations by Wilcox (2001) have serious implications for feature selection methods based on Pearson's correlation coefficient. First of all, highly predictive features may be discarded, simply because the presence of outliers causes the computed sample correlation coefficient to be small, or worse still, to be insignificant. Secondly, non-linear associations will produce very small correlation coefficients, which will cause otherwise relevant features to be discarded. In a nutshell, in the presence of outliers and non-linear associations, and this should be expected in data mining, Pearson's correlation coefficient will provide a feature ranking which is incorrect. When this is the case, there is an increased risk of creating a model that has poor predictive performance. The reader will recall that the use of poor predictors results in an increase in the inherent error or variance component of the prediction error. Wilcox (2001) has made a strong point that*: 'if we are told r (Pearson's sample correlation coefficient), and nothing else, we cannot deduce much about the details of how X and Y are related'.* A third problem with Pearson's correlation (and other correlation measures) is that the two random variables *X* and *Y* may be strongly associated for some of the values and not for the whole range of values. When this is the case, computing the correlation coefficient between *X* and *Y* based on the whole range of each of the variables, will provide very small correlation coefficient values which will lead to the assumption that there is no association between the variables.

3.5.2 Robust measures of correlation

Wilcox (2001) has discussed three ways of handling outliers when computing sample correlations. The first method is to compute a winsorised Pearson's correlation coefficient, the second method is to use Spearman's *rho* correlation coefficient, and the third method is to use Kendall's *tau* correlation coefficient. To winsorise the values of a random variable *X*, the smallest *z%* and largest *z%* of values in the sample are altered. The alteration involves replacing each of the small values with the smallest of the unaltered values, and replacing the large values with the largest unaltered value (Wilcox, 2001). For correlation computations, the values of both *X* and *Y* must be winsorised, prior to computing the sample means, variances and covariance. The problem with computing the trimmed means and winsorised variances is that the values of the variables must be sorted first. For a multivariate dataset with *d* variables $2d + d^2$ sorting operations must be conducted for the computation of the class-feature and feature-feature correlations. These intensive computations can be avoided by using Spearman's *rho* or Kendall's *tau* correlation coefficients.

For Spearman's *rho*, the values of *X* and *Y* are converted to ranks, *1,..n*. Spearman's *rho* is then computed with Pearson's correlation formula using the rank values. This way, the effect of outliers is avoided. Even though this method eliminates the problem of outliers, its computational requirements are not modest. It is still necessary to sort the values of *X* and *Y*. For multivariate data, $2d + d^2$ sorting operations are still needed for the *X,Y* pairs. For Kendall's *tau* computations are needed for the probabilities π_c and π_{d}. π_c is the probability that, given two random pairs of values (x_1, y_1) and (x_2, y_2), when $x_1 > x_2$ then $y_1 > y_2$, and when $x_1 < x_2$ then $y_1 < y_2$. π_c is called the probability of *concordance* between the random variables *X* and *Y*. π_d is the probability that the opposite is the case, and is called the *discord*ance between *X* and *Y*. The value of Kendall's *tau* is computed as $\tau = \pi_C - \pi_D$. The probabilities π_c and π_d are estimated by comparing all possible sets of pairs of values of the variables *X* and *Y*, that is, $(n-1)/2$ pairs. Kendall's *tau* is computationally more efficient than Spearman's *rho* since Kendall's *tau* does not require sorted data. The method is also a good alternative to Pearson's coefficient when outliers and non-linearity are present. However, the computational time complexity for Kendall's *tau* is still quadratic in *n*, the size of the sample used to estimate the correlations.

3.6 Validation methods for feature selection

Guyon and Elisseeff (2003) have defined feature validation methods as those methods that are used to determine the number of significant features, guide and halt the feature subset search or, evaluate the final performance of the models based on the selected features. The discussion in this section is concerned with methods for determining the validity of the decision to select a given feature for inclusion in the set of predictive features. Section 3.6.1 discusses the need for validation of class-feature and feature-feature correlation coefficients. The practical significance of correlation coefficients is discussed in section 3.6.2. Validation methods based on hypothesis testing and based on fake variables are respectively discussed in sections 3.6.3 and 3.6.4.

3.6.1 The need for validation of correlation coefficients

In general, filtering methods rank features based on the correlation or some other measure, with the class. The higher the measure the more predictive a feature is assumed to be. Smyth (2001) has argued that a large correlation coefficient between the random variables *X* and *Y* in a given data sample could be purely due to chance. If the feature-class correlations were measured on a different sample, the correlation coefficient would take on different values. The same argument applies to any measurement that is taken on a data sample. Measures such as the class-feature correlation coefficient $corr_{cf}$ and the feature-feature correlation coefficient $corr_{ff}$, which appear in equations (3.1), (3.2) and (3.3) are therefore random variables with associated probability distributions. It is essential to establish the validity of these correlation measures before they are used in feature ranking and subset selection. One validation method for feature correlations that has been reported in the literature is the use of fake variables (Guyon & Elisseeff, 2003; Bi et al, 2003; Stoppiglia et al, 2003). A fake variable or probe, is a variable whose values are generated purely at random. Such values should not have any correlation with the class variable. When measuring correlations using either Pearson's *r* or Kendall's *tau*, any features with a correlation value that is lower than that of the fake variables should be discarded.

3.6.2 Practical significance of correlation coefficients

It was pointed out in section 3.1 that feature relevance and redundancy are typically defined in terms of the strength of correlations. Blum and Langley (1997) have defined a relevant feature as one which is *highly* correlated with the class variable. Hall (1999) and Ooi et al (2007) have defined a redundant feature as one that is *highly* correlated with other features. Cohen (1988) has observed that different fields of study and research define the quantitative adjectives for correlations, namely *small, medium, large*, differently. In the physical sciences where the variable values are obtained from high precision instruments, a correlation coefficient of *0.9* is considered small (Cohen, 1988). In Economics, a correlation coefficient of *0.6* is considered small (Coetsee, 2007).

For the field of Behavioural Sciences research, Cohen (1988) has suggested the following approach to interpreting the magnitude of a correlation. A value in the range [*0.10*, *0.30*) indicates a small/weak correlation. A value in the range [*0.30*, *0.50*) indicates a medium correlation. A value in the range [*0.50*, *1.00*] indicates a large/strong correlation. Cohen (1988) has argued that these criteria are suitable for the social sciences, since for this field of research there is always a large number of complicating factors in the experimental setup and the measuring instruments used to collect data. One implication of Cohen's (1988) criteria for interpreting correlations is that correlation values of less than *0.10* have no practical significance, even though such correlation coefficients might appear to be statistically significant, especially for very large samples.

3.6.3 Validation based on hypothesis testing for correlation coefficients

In many fields of scientific enquiry, it is common practice to establish the statistical significance of the correlation coefficient, *r*, using Student's *t-test* for correlations (Wilcox, 2001; Kanji, 1999). One can then test the null hypothesis: H_0*: 'the correlation between the two variables is zero',* and the alternative hypothesis H_a *: 'the correlation between the two variables is not zero'.* If the null hypothesis H_0 is rejected, then one concludes that the two variables have a statistically significant (linear) relationship indicated by the direction and magnitude of the correlation coefficient *r*. The *T* statistic used for testing H_0 and H_a as defined above is

$$T = r\sqrt{\frac{n-2}{1-r^2}} \qquad (3.7)$$

where *n* is the sample size used to estimate *r*. Under normality and when the population correlation $\rho = 0$ the quantity *T* has a Student's *t* distribution with *n-2* degrees of freedom. Even though this is a fairly popular test in many research areas, the author is not aware of any reported usage of this test in feature selection for computational data mining.

To test the hypothesis $H_0 : r = c$, that is, the correlation coefficient is some value *c* other than zero, Fisher's transform is used to convert the correlation coefficients into the *Z* statistic as follows (Cohen, 1995; Cohen, 1988):

$$Z = \frac{Z(r) - Z(c)}{\sigma_{z(r)}} \tag{3.8}$$

In equation (3.8) *Z(r)* is Fisher's *Z* transform of *r* and is computed as

$$Z(r) = \tfrac{1}{2} \ln\left(\frac{1+r}{1-r}\right) \tag{3.9}$$

where $\sigma_{z(r)}$ is the estimated standard error of the sampling distribution of *Z(r)* and is computed as $1/\sqrt{(n-3)}$, where *n* is the number of values used to compute *r*. *Z(c)* is Fisher's Z transform of *c* and can be similarly computed.

3.6.4 Validation based on fake variables

Stoppiglia et al (2003) have proposed the use of probes (fake variables) for determining the cut-off point between relevant and irrelevant features. A probe is a random variable whose values may be generated from any probability distribution. Stoppiglia et al (2003) have argued that, since such a random variable should not have any significant correlation to the target function, it should be ranked last if an infinitely large amount of data were available. However, since the amount of data is finite, the probe should appear somewhere in the ranked list and all features that are ranked below the probe should be discarded. Since the probe is a random variable, its rank in the list of features is also a random variable. The decision to keep or discard features based on the probe's value should be based on the probability that this feature's ranking is higher or lower than that of the probe. Stoppiglia et al (2003) have recommended that the designer of the model should choose a tolerable risk of selecting or discarding the feature based on the ranking of the probes.

Bi et al (2003) have reported studies on feature selection for micro-array datasets. Bi et al (2003) have observed that the feature selection process can be very unstable in the sense that each time a feature set is selected it consists of totally different features. Since micro-array datasets are typically small, Bi et al (2003) have used bootstrap samples and merged the results from these samples. Bi et al (2003) have used 3 fake variables drawn randomly from Gaussian distributions and have discarded all variables that are less relevant than one of the fake variables. It should be noted however, that since fake variables are also random variables, one should expect that the ranking of the fake variable will vary from sample to sample.

3.7 Conclusions

The need for feature selection for predictive data mining has been discussed in this chapter. Feature selection methods for implicit and explicit feature selection have been presented. Implicit feature selection is performed by the induction algorithm, while explicit feature selection is performed by an algorithm whose sole purpose is to select the best features for a given prediction task. Methods for measuring class-feature correlations have been discussed and the problems inherent in these methods have been highlighted.

Filtering methods are heavily dependent on the class-feature and feature-feature correlation measures. Many researchers have used Pearson's correlation coefficient to establish class-feature and feature-feature correlations. Even though this is the most reliable and efficient way of measuring linear correlations, it is not the most appropriate measure when correlations are non-linear, and when outliers are present. It is useful to study more robust measures of correlation for feature selection.

The validation methods for selected feature subsets that have been reported are based on the use of fake variables. These methods have been studied in the domain of micro-array datasets, where the datasets are typically small: less than 200 instances. Given that fake variables are random variables, it is useful to conduct studies on how probes will perform in the presence of much larger datasets, and to devise methods of using probes to select reliable feature subsets. In many fields of research, hypothesis testing is used to establish the statistical significance of a correlation coefficient. There are no reported studies of this nature for feature selection for computational predictive data mining. It is the author's belief that, when large amounts of data are available, opportunities arise for researchers to conduct studies of this nature.

Filtering methods conduct feature subset selection based on a general definition of relevance, redundancy and correlation strength, even though different application domains have different interpretations of what it means for two variables to be highly correlated. Algorithms are needed that can incorporate domain-specific definitions of feature relevance and redundancy. Even though many studies have been reported on feature selection for small datasets, to the author's knowledge there are very few reported studies (e.g. Liu & Setiono, 1998) that specifically address feature selection from very large datasets. It is the author's opinion that it is useful to conduct more studies of feature selection methods that can make use of the large amounts of available data in order to perform reliable measurement and validation of class-feature correlations and as a result, provide reliable feature subsets.

Chapter 2 presented a discussion of current methods of training dataset selection from large datasets. It was argued that in the presence of very large datasets it is useful to conduct studies of dataset selection methods aimed at reducing the bias and variance components of the prediction error, without having to re-use the training data. One method of reducing variance errors is the selection of a good set of predictive features. In this chapter, current methods of feature selection have been discussed and it has been argued that it is useful to conduct studies of feature selection methods that make use of the large amounts of available data to perform reliable measurements and validation of class-feature correlations. The next chapter presents a discussion of the research methods used for the studies in this thesis. The studies on feature selection methods are presented in chapter 5. The studies on training dataset selection are presented in chapters 6, 7, 8 and 9.

Chapter 4

Research Methods

> ‘It is better to have an approximate answer to the right question than a precise answer to the wrong question which can be made…precise’ (John Tukey)

The discussion in chapters 2 has made it clear that, first of all, it is necessary to conduct training dataset selection from large datasets for purposes of computational efficiency. Secondly, it is beneficial to study methods for selection of training data based on the characteristics of the instance space. Thirdly, the point has been made that the use of aggregate models has the potential to increase predictive accuracy since aggregate models are aimed at the reduction of the variance component of the prediction error. The use of training dataset selection methods aimed at the reduction of the bias and variance components of the prediction error should result in predictive models with a higher level of performance, compared to models created from data selected purely at random. The discussion of chapter 3 has made it clear that many samples should be used for the measurement and validation of the correlations for the dataset features in order to ensure reliable feature selection for large datasets.

The purpose of this chapter is to explain how methods that address the above issues were studied. Detailed discussion of the research questions and objectives, the central argument of the thesis and, the research paradigm that was followed, are given in sections 4.1, 4.2 and 4.3 respectively. The datasets used for the experiments and the sampling procedures used are discussed in sections 4.4 and 4.5 respectively. The data mining algorithms used in the experiments, the methods used to evaluate model performance and, the software used for the experiments are given in sections 4.6, 4.7 and 4.8 respectively. Section 4.9 gives a summary of the chapter.

4.1 Research questions and objectives

The discussions in chapters 2 and 3 led the author to pose the following main research question:

What methods of training dataset selection can be used to obtain as much information as possible from large datasets while at the same time using training datasets of small sizes to create predictive models that have a high level of predictive performance?

Based on the main research question and the literature review given in chapters 2 and 3, the primary objectives for conducting the research were to investigate promising methods for the following:

(1) Reliable feature selection from large datasets using as much data as is feasible.

(2) Design of aggregate models which can make use of large amounts of training data while avoiding the problem of modeling phantom structure (i.e. structure that occurs purely due to chance as discussed in section 2.8.2).

(3) Design and selection of training datasets for base models aimed at increasing predictive accuracy through the reduction of bias and variance of the prediction error.

(4) Creation of a theoretical model that helps to explain the factors that affect the quality of selected features and the relationships between these factors.

(5) Creation of a theoretical model that helps to explain the factors that affect the performance of aggregate models and the relationships between these factors.

4.2 The central argument for the thesis

The central argument of this thesis is that, for predictive data mining, it is possible to systematically select many dataset samples and employ different approaches (different from current practice) to feature selection, training dataset selection, and model construction. When a large amount of information in a large dataset is utilised in the modeling process, the resulting models will have a high level of predictive performance and should be more reliable.

Feature selection has been traditionally done based on a single randomly selected sample of the data. In the presence of very large amounts of data, many samples can be used in order to more reliably measure and validate the correlations between the potential predictive features and the class (predicted) variable. The construction of base models that make up an aggregate model requires the use of a separate training dataset for each base model. It has been argued that syntactic diversity in the base models is a key factor in increasing aggregate model performance. For small datasets it is difficult to create sufficiently large training datasets that are also highly diverse. Breiman (1996) has attempted to achieve diversity through bootstrap sampling. With large amounts of data, it is easier to create diverse training datasets, since there is plenty of data to choose from. Through the use of boosting, Freund and Schapire (1997) have attempted to replicate the training instances that come from those regions of the instance space that are difficult to predict correctly. With large amounts of data, it is easier to obtain many instances that come from the difficult regions. Very large datasets provide far better coverage of the instance space, compared to small datasets. Examination of the structure of the instance space should lead to a better understanding of the prediction task at hand. This understanding should lead to better decisions for the sample composition of the training datasets for base models.

4.3 The research paradigm and methodology

The research paradigm used for this research is design science research as described by March and Smith (1995), Hevner et al (2004), Vaishnavi and Kuechler (2004/5) and Manson (2006). In this section, the design science research paradigm and methodology are briefly discussed. The design science research paradigm and the outputs of design science research are respectively presented in sections 4.3.1 and 4.3.2. Artifact evaluation and theory building, and the justification for adopting design science research for this thesis are respectively discussed in section 4.3.3 and 4.3.4. The different types of theories for data mining are discussed in section 4.3.5.

4.3.1 The design science research paradigm

The design science research paradigm originates from engineering and the physical sciences (March & Smith, 1995). Design science (Simon, 1996) and design science research (March & Smith, 1995) are concerned with the design and study of artifacts.

Hevner et al (2004) have provided the following definition for Information Systems artifacts:

> '.. innovations that define ideas, practices, technical capabilities, and products, through which the analysis, design, implementation, and use of information systems can be effectively and efficiently accomplished.'

Design science research involves two distinct steps as depicted in figure 4.1. In the first step, an artifact is created. In the second step, an analysis of the usage and performance of the artifact is conducted. The purpose of the analysis is to understand, explain, and possibly improve on one or more aspects of the artifact (Vaishnavi & Kuechler, 2004/5). According to Hevner et al (2004), in the context of information systems, artifacts may be models (abstractions and representations), methods (algorithms and practices) and instantiations (implemented and prototype systems). Design science research is a problem solving paradigm which seeks to create innovations in terms of ideas, practices, technical capabilities, and products. Through these innovations, the analysis, implementation, and usage of information systems can be effectively accomplished. Another view of design science research is due to March and Smith (1995). March and Smith (1995) have defined design science research and design science as activities aimed at the creation of things that serve human purposes. Design science and design science research are therefore technology-oriented and their outputs are assessed against criteria of value/utility.

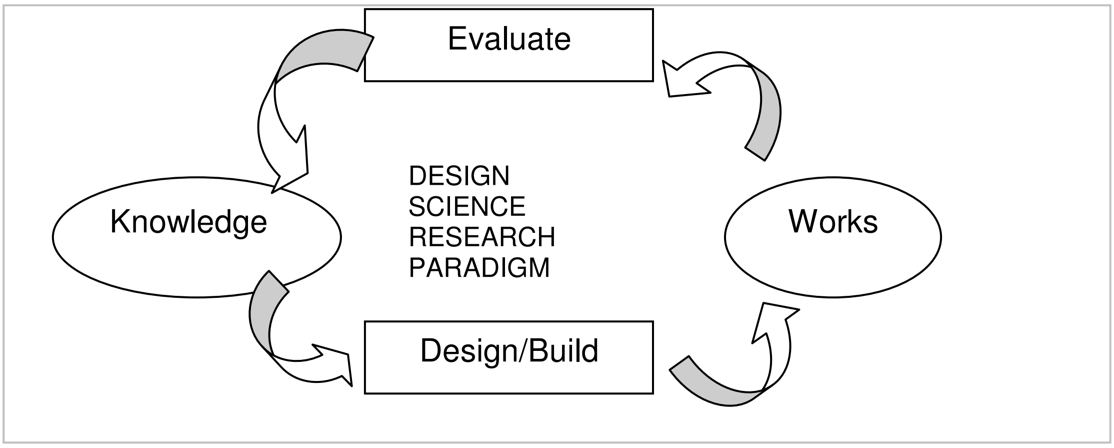

Figure 4.1: A general model for generating knowledge in design science research (adopted from Vaishnavi & Keuchler (2004/5) and Manson (2006))

Manson (2006) has summarised these views by observing that design science research is a process of using knowledge to design and create useful artifacts, and then using rigorous methods to analyse why, or why not, a particular artifact is effective. Figure 4.1 shows a general model for generating knowledge in design science research, and reflects Manson's (2006) observations.

4.3.2 The outputs of design science research

Table 4.1 gives a list of the outputs of design science research. Scientific research is about generating knowledge. In terms of generating new knowledge, for design science research new knowledge is generated in terms of the new constructs, new models, new methods (the how-to knowledge), and the better theories that arise out of the design and evaluation activities. Constructs are the core vocabulary that is used to express the concepts of a field. Knowledge is created when statements or propositions are made to express the relationships between various constructs of the field. Better theories, in terms of the models, will result if the models are rigorously tested in order to establish the existence of the relationships.

Table 4.1: Outputs of design science research: Adapted from Vaishnavi & Kuechler (2004/5)

	Output	Description
1	constructs	Conceptual vocabulary of a domain. Constructs make up the language used to define and communicate problems and solutions.
2	models	A set of propositions or statements expressing relationships between constructs
3	Methods	a set of steps used to perform a task: **how-to knowledge**
4	Instantiations	Operationalisation of constructs, models and methods. Demonstration that the models and methods can be implemented in a working system.
5	**Better theories**	Artifact construction as analogous to experimental natural science

4.3.3 Artifact evaluation and theory building

March and Smith (1995) have defined theories as 'deep, principled explanations of phenomena'. Cohen (1995) has argued that theories may also be 'propositions from which we can derive testable hypotheses'. Table 4.1 shows that one of the outputs of design science research should be 'better theories', that is, some improvements should be made to the existing theories of the field. Cohen (1995:ch.9) has provided guidelines for generalisation and theory building in Artificial Intelligence (AI) research. Cohen (1995) has stated that, for AI research there are six possible types of contributions as shown in figure 4.2. The cells 3,4,5,6 (P-S, P-G, E-S, E-G) in figure 4.2 represent research activities that result in the creation of new scientific theories. Cells 3 and 4 (P-S and P-G) represent the creation of predictive theories. Predictive theories attempt to predict (hypothesize on) the behaviour of a specific system or a class of systems. According to Cohen (1995), system behaviour is typically predicted in terms of the features of the system architecture (structure), the features of the tasks that the system can perform, and the features of the environment in which the system operates. Cells 5 and 6 (E-S and E-G) represent the creation of explanatory theories. Explanatory theories attempt to explain the hypothesized behaviour of a specific system or class of systems.

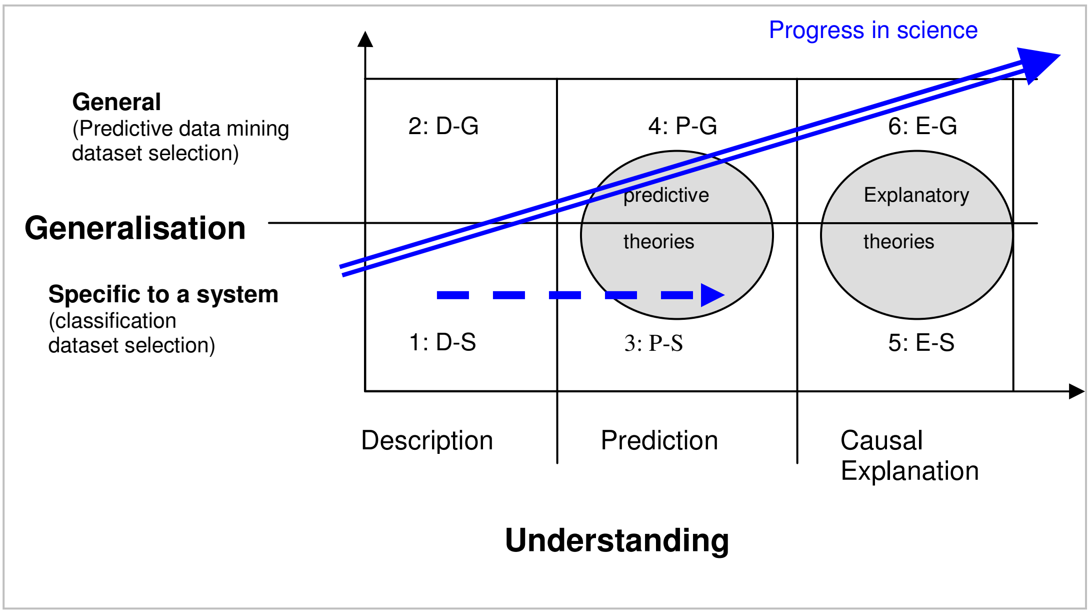

Figure 4.2: Relationship between understanding, generalisation and scientific theories. Adapted from Cohen (1995)

According to Cohen (1995) progress in science is gradually achieved by moving from descriptions of specific systems to providing causal explanations for systems in general as depicted in figure 4.2. Specific to design science research, March and Smith (1995) have observed that progress in design science is achieved when existing technologies are replaced by more effective ones. For the scope of this research, 'general' systems were viewed as systems for dataset selection for predictive data mining. A system for dataset selection for discriminative classification modeling was viewed as a 'specific' system as depicted in figure 4.2. The dashed line in figure 4.2 indicates the scope of scientific progress addressed in this thesis based on Cohen's (1995) definitions. The scope of design science progress claimed in this thesis is described in detail in chapter 11.

In the process of formulating predictive and explanatory theories, the Scientific Method of Peirce and Popper (Ngwenyama & Osei-Bryson, 2010; Oates, 2006) may be followed for purposes of building theories based on empirical studies. This method involves observation, hypothesis generation, experiment design, and testing the validity of the hypotheses. Figure 4.3 shows the steps of the scientific method based on the discussion by Ngwenyama and Osei-Bryson (2010). Empirical observation involves the gathering of data/information about the phenomenon of interest. Hypothesis generation is when the researcher puts forward several hypotheses that could explain the phenomenon. Experiment design involves the design of experiments to test the logical consequence (validity) of the hypotheses. In the empirical testing step, the experiments are conducted in order to collect observations/data which is then analysed in order to establish whether or not the hypotheses are valid. The building of new theories arises from the outcomes of the empirical testing step. The empirical research reported in this thesis resulted in the formulation of a number of predictive theories which are presented in chapter 11. The scientific method was followed in the design and evaluation steps within the design science research paradigm.

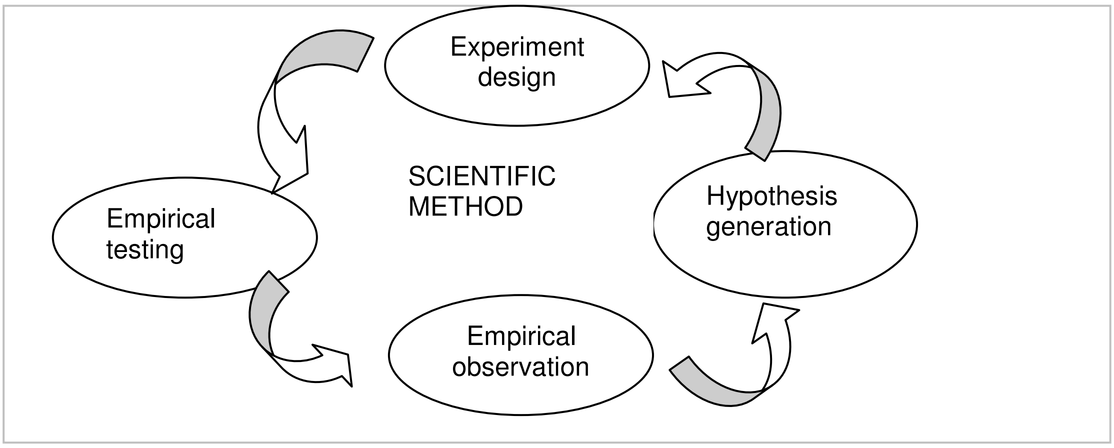

Figure 4.3: Steps of the scientific method.

4.3.4 Justification for adopting the design science research paradigm

This thesis is concerned with the investigation of methods for selecting features and training datasets from large amounts of data and the use of the selected data to create predictive models which can achieve a high level of accuracy and stability. The design and evaluation activities for the research are therefore the design and evaluation of feature selection methods, training set selection methods, and associated methods for model construction and testing. The use of design science research enabled the author to systematically evaluate the hypothesised methods of feature selection, dataset selection, and model construction, and to systematically test hypothesised relationships between factors that affect the quality of selected features and training datasets. Based on existing literature, the author was able to construct models of known factors that affect the quality of selected features and training datasets, and to extend these models based on the results obtained from the experiments that were conducted for this research.

4.3.5 Theories for data mining

The foregoing discussion is aimed at the development of empirically derived theories for data mining. It was noted in chapter 1 that the parent fields of data mining are Computer Science and Statistics (Smyth, 2001). More recently, Olafsson et al (2008) have discussed the contributions by Operations Research to the field of data mining. While Statistics and Operations Research are founded on mathematical theories, in general for Computer Science, there are two types of possible theories: mathematical theories and empirical theories (Simon, 1996). Simon (1996) has observed that there are many aspects of computer systems that are so complex that there are no feasible mathematical theories that can be developed to describe their design and behaviour. Specific to machine learning, there are many mathematical theories that have been developed. However, Dietterich (1997) has observed that many problems in machine learning will only be solved through empirical studies, and not through mathematical formulations. The theories discussed in chapter 2, on sample complexity for inductive algorithms are a case in point. It was stated in chapter 2 that these theories provide unrealistic estimates for the sample complexity. Cohen (1995) has provided comprehensive guidelines on how to conduct empirical research in artificial intelligence and how to generate empirical theories from the empirical studies.

4.4 The datasets used in the experiments

The datasets used for the experiments were obtained from the UCI KDD Archive (Bay et al, 2000; Hettich & Bay, 1999), and the UCI Machine Learning Repository (Ascuncion & Newman, 2007; Blake & Merz, 1998). This section provides the motivation for the choice of datasets, brief descriptions of the datasets, past usage of the datasets, and pre-processing that was performed on two of the datasets. The descriptive statistics for the selected datasets are provided in appendix C. The reasons for selecting the datasets for the experiments of this thesis are presented in section 4.4.1. Dataset pre-processing is discussed in sections 4.4.2 and 4.4.3.

4.4.1 Choice of datasets and past usage

Typical empirical studies on aggregate modelling for small datasets have been conducted using small numbers of datasets. The exception is Ali and Pazzani's (1996) studies where 29 datasets have been used. Table 4.2 provides some examples of studies where small datasets have been used. Experimental studies on dataset selection and aggregate modelling from large datasets have also been conducted using small numbers of datasets.

Table 4.2: Examples of datasets used in data mining and machine learning studies

Author(s)	Nature / scope of study	Dataset description
Ooi et al (2007)	OVA classification and feature selection	8 small datasets of sizes between 60 and 257 instances, and large number of dimensions ranging between 1,741 and 12,011
Chawla et al (2001)	Dataset partitioning and aggregate modeling using massively parallel super computers	4 small datasets of size less than 20,000 instances and **2 large datasets of** size 299,186 **and** 3.6 million instances
Hall et al (2000)	Dataset partitioning and aggregate modelling using massively parallel super computers	**4 very large datasets** of sizes 1.6, 3.2, 6.4 and 51 million instances
Chan and Stolfo (1998)	Dataset partitioning, sampling and aggregate modelling	**1 large dataset** for credit card fraud detection. 500,000 instances
Ho (1998)	Dataset partitioning with random feature subsets	4 small datasets of sizes between 3,186 and 14,500 instances
Ali and Pazzani (1996)	Factors that affect performance improvements for aggregate models	29 small datasets of sizes between 150 and 8,200 instances
Breiman (1996)	Bootstrap aggregation for classification and regression	12 small datasets of sizes between 351 and 1,395 instances
Kwok and Carter (1990)	aggregate modeling for decision trees	2 small datasets of sizes 446 and 5,516 instances

The examples given in table 4.2 indicate that studies have been conducted using one, two or four large datasets. Studies on extremely large datasets with instances in excess of one million have been conducted using supercomputers with massively parallel architectures (Chawla et al, 2001; Hall et al, 2000). Based on the foregoing observations and the time and computational resources available to the author, a decision was made to use two small datasets and two large datasets for the experiments for this thesis.

Table 4.3 gives the characteristics and past usage of the datasets used in the experiments for this thesis. The mushroom and abalone datasets (Ascuncion & Newman, 2007; Blake & Merz, 1998) are very commonly used in machine learning research. The wine quality dataset (Cortez et al, 2009; Ascuncion & Newman, 2007) has been used by Cortez et al (2009) for predictive modeling of wine quality. The mushroom dataset has originally been used for concept learning research by Schlimmer (1987) and Iba et al (1988). The mushroom dataset was selected for this research as the dataset which consists of only qualitative features, in order to study the behaviour of correlation measures for qualitative features. The abalone dataset has originally been used by Waugh (1995) for cascade-correlation. The original abalone dataset has 29 classes. Clark et al (1996) have created a three-class version of this dataset for their comparative study of artificial neural network algorithms. The three-class version of the abalone dataset was used for the experiments reported in this thesis. The wine quality dataset was selected for purposes of establishing whether the proposed training instance selection methods can also be applied to small datasets. A second reason for selecting the abalone and wine quality datasets was because these datasets have low levels of classification accuracy and can therefore be used to demonstrate increases in predictive performance (Cohen, 1995).

The two large datasets that were used for the experiments are forest cover type and KDD Cup 1999 (Bay et al, 2000; Hettich & Bay, 1999). These two datasets were chosen for the experiments because they are large datasets, and have large numbers of features. Tables 4.3, 4.4 and 4.5 show the important statistics for these datasets. The forest cover type dataset is a good example of data mining for a scientific application. This dataset consists of data describing the forest cover type for each of 581,012 forest cells, each measuring 30x30 meters. The prediction task for the forest cover type dataset is to predict one of seven forest cover types based on the soil type, wilderness area type, elevation (altitude) and other variables. Blackard (1998) has used this dataset to study the differences in predictive performance between artificial neural networks and discriminant analysis. The KDD Cup 1999 dataset is a typical example of data for forensic data mining. This dataset is a common benchmark for the evaluation of computer network intrusion detection systems (IDS). The dataset consists of a wide variety of network intrusions, simulated for a military computer network environment.

Table 4.3: The datasets used for the experiments

Dataset	Description		Past usage
	Size	Features & classes	
Forest cover type	581,012	54 features 10 continuous, 44 binary, 7 classes	Comparison of ANNs and discriminant analysis (Blackard, 1998)
KDD Cup 1999 Training dataset (10% version)	494,022	41 features 34 continuous, 7 qualitative, 23 classes	Intrusion detection (Stolfo et al, 2000)
KDD Cup 1999 Test dataset	311,029	41 features 34 continuous, 7 qualitative, 40 classes	
Wine quality (white)	4,898	11 continuous-values features, 7 classes	Prediction of wine quality scores assigned by wine tasters. (Cortez et al, 2009)
Abalone (3 class)	4,177	features: 8 features 7 continuous, 1 qualitative, 3 classes	Prediction of the age of abalone
Mushroom	8,146	22 qualitative features 2 classes	Prediction of edibility of mushrooms

Table 4.4: Class counts for the forest cover type dataset

Class	Type of forest cover	Count
1	Spruce / Fir	211,840
2	Lodgepole pine	283,301
3	Ponderosa pine	35,754
4	Cottonwood / Willow	2,747
5	Aspen	9,493
6	Douglas - fir	17,367
7	Krummholz	20,510
TOTAL		**581,012**

The dataset was provided by the USA DARPA and MIT Lincoln Labs (Lee et al, 2002), and was later pre-processed for the KDD Cup 1999 competition by the Columbia IDS Group (Stolfo et al, 2000). Two versions of this dataset are provided at the UCI KDD archive. The smaller version, which consists of ten percent of the instances of the original version, was used for the experiments. Four main categories of attacks are present in the dataset: denial-of-service (DOS), unauthorized access from a remote machine (R2L), unauthorized access to super-user privileges (U2R), and probing attacks (PROBE) (Laskov et al, 2005; Lee & Stolfo, 2001; Stolfo et al, 2000).

Table 4.5: Class counts for the KDD Cup 1999 training (10% version) and test sets

	Class count			Class count	
Attack Type	Training set	Test set	AttackType	Training set	Test set
apache2		794	portsweep	1,040	354
back	2,203	1,098	processtable		759
buffer_overflow	30	22	ps		16
ftp_write	8	3	rootkit	10	13
guess_passwd	53	4,367	saint		736
httptunnel		158	satan	1,589	1,633
imap	12	1	sendmail		17
ipsweep	1,247	306	smurf	280,790	16,4091
land	21	9	snmpgetattack		7,741
loadmodule	9	2	snmpguess		2,406
mailbomb		5,000	spy	2	
mscan		1,053	sqlattack		2
multihop	7	18	teardrop	979	12
named		17	udpstorm		2
neptune	107,201	58,001	warezclient	1,020	
nmap	231	84	warezmaster	20	1,602
normal	97,278	60,593	worm		2
perl	3	2	xlock		9
phf	4	2	xsnoop		4
pod	264	87	xterm		13
			TOTALS	**494,021**	**311,029**

4.4.2 Dataset pre-processing to balance class distributions

The KDD Cup 1999 dataset is not amenable to classifier construction without pre-processing (Laskov et al 2005; Leung & Leckie, 2005). Laskov et al (2005) have observed that the KDD Cup 1999 dataset suffers from two major flaws in the distribution of the classes in the dataset. First of all, approximately 80% of instances correspond to attacks. Secondly, the distribution of the attack instances is highly unbalanced. Laskov et al (2005) have observed that *Probes* and *DOS* attacks dominate the class distribution, while more dangerous attacks such as *phf* and *imap* are severely under-represented. Researchers who have used the KDD Cup 1999 dataset (e.g. Shin & Lee, 2006; Laskov et al, 2005; Leung & Lecki, 2005) have typically pre-processed the dataset to balance the distribution of the attack types and service types, and to reduce the number of instances for attacks in comparison to normal connections. Laskov et al (2005), for example, have reduced the number of attack instances to five percent (5%).

Table 4.6: Reduction of the over-representation of (service, attack type) values in the KDD Cup 1999 training and test datasets

Dataset	Service name	Class name	Instance type	Count before	Count after
Training set	private	neptune	attack	101,317	500
	ecr_i	smurf	attack	280,790	1,000
	http	normal	normal	61,887	5,000
	smtp	normal	normal	9,598	5,000
Test set	private	neptune	attack	54,739	500
	private	smurf	attack	164,091	1,000
	private	snmpgetattack	attack	7,733	2,500
	smtp	mailbomb	attack	5,000	2,500
	private	normal	normal	12,808	2,500

Two approaches have been used by researchers to construct classification models from the KDD Cup 1999 dataset. For the first approach, the attack types that appear in the data are used as the classes (Laskov et al, 2005; Lee & Stolfo, 2000; Fan et al, 2000). For the second approach, the main categories: NORMAL, DOS, PROBE, R2L and U2R are used as the classes (Shin & Lee, 2006). The main problem with using the first approach is that there are attack types that are severely under-represented as can be seen in table 4.6. Secondly, there are attack types that appear in the test set but not in the training set. The problem of under-representation is slightly reduced in the second approach. The problem of classes which appear in the test set and not the training set is partially eliminated when the second approach is used, since all such attacks belong to the R2L category.

For the experiments reported in this thesis, pre-processing of the dataset was done as follows. The instances for the *(service, attack type)* values that are severely over-represented were reduced as shown in table 4.6. The objective of the reduction was to ensure that the frequency of attacks for each over-represented attack type is reduced to make that frequency comparable to the other attack types for that service. The reduction was achieved using sequential random sampling of the instances that are over-represented. The training and test datasets were further pre-processed to add a new class variable with values NORMAL, DOS, PROBE, R2L and U2R. Table 4.7 shows the resulting attack type and class distributions after the pre-processing, for the dataset used for training. The test dataset was further pre-processed to remove all attack types that do not appear in the training data. This was motivated by the following observations as stated by Lee and Stolfo (2000).

The two main intrusion detection techniques are *misuse detection* and *anomaly detection*. Misuse detection systems use patterns of well known attacks to identify known intrusions. On the other hand, anomaly detection systems detect and report activities that significantly differ from established normal usage profiles (Lee & Stolfo 2000). Since classification modeling is based on inductive learning, classification models created for intrusion detection systems should be created for misuse detection. For this reason, attack types that do not appear in the training data were removed from the test data. Table 4.8 shows the resulting class distribution of the test dataset after this phase of pre-processing. The entries for TestA in column 7 of table 4.8 indicate the number of instances that were removed because the attack type does not appear in the training data.

Table 4.7: Class counts for the final version of the KDD Cup 1999 training dataset

Class	Type of connection	AttackType	AttackType count	Class count
DOS	Denial of service	back	2,203	
		land	21	
		neptune	6,384	
		pod	264	
		smurf	1,000	
		teardrop	979	10,851
NORMAL	normal	normal	35,794	35,794
PROBE	Probing prior to attack	ipsweep	1247	
		nmap	231	
		portsweep	1,040	
		satan	1,589	4,107
R2L	Unauthorised access from a remote machine	ftp_write	8	
		guess_passwd	53	
		imap	12	
		multihop	7	
		phf	4	
		spy	2	
		warezclient	1,020	
		warezmaster	20	1,126
U2R	Unauthorised access to superuser privileges	buffer_overflow	30	
		loadmodule	9	
		perl	3	
		rootkit	10	52
TOTALS			**51,930**	**51,930**

Table 4.8: Class counts for the final version of the KDD Cup 1999 test dataset

Class	AttackType	AttackType count	Class count	Class	AttackType	Attack count	Class count
DOS	apache2	794	10023	R2L	httptunnel	TestA: 158 TestB: 0	TestA: 11,114 TestB: 5,995
	back	1,098			imap	1	
	land	9			multihop	18	
	mailbomb	2,500			named	TestA: 17 TestB: 0	
	neptune	3,762			phf	2	
	pod	87			sendmail	TestA: 17 TestB: 0	
	processtable	759			snmpgetattack	TestA : 2508 TestB: 0	
	smurf	1,000			snmpguess	TestA: 2406 TestB: 0	
	teardrop	12			warezmaster	1,602	
	udpstorm	2			worm	2	
NORMAL	normal	50,285	50285		xlock	TestA: 9 TestB: 0	
PROBE	ipsweep	306	4166		xsnoop	TestA: 4 TestB: 0	
	mscan	1,053		U2R	buffer_overflow	22	70
	nmap	84			loadmodule	2	
	portsweep	354			perl	2	
	saint	736			ps	16	
	satan	1,633			rootkit	13	
R2L	ftp_write	3			sqlattack	2	
	guess_passwd	4,367			xterm	13	
				TOTALS (TestA)		**75,658**	**75,658**
				TOTALS (TestB)		**70,539**	**70,539**

4.4.3 Dataset pre-processing to normalise feature values

The KDD Cup 1999 dataset contains features from various numeric-valued domains. Table 4.9 shows a selected sample of features as well as the minimum and maximum values of the features for the KDD Cup 1999 dataset. As can be seen in table 4.9, the KDD Cup 1999 dataset has features with a narrow value range (e.g. [0,1] for *DstHostSrvSerrorRate*) as well as features with a very wide value range (e.g. [0, 693375640] for *SrcBytes*). K-Nearest neighbour (KNN) is one of the classification algorithms that were used in the experiments. The KNN algorithm computes distances between instances using a distance measure from the class of the Minkowski norms (Doherty et al, 2007) of which the Euclidean distance measure is the most common. The computation of the Euclidean distance using features from very widely varying ranges of values such as found in the KDD Cup 1999 dataset will result in the large-valued features dominating the result of the computed distance, and so masking the effects of the small-valued features.

Table 4.9: Range of values for features in the KDD Cup 1999 dataset

Feature	**Minimum value**	**Maximum value**
NumCompromised	0	884
WrongFragment	0	3
DstHostSrvSerrorRate	0	1
Hot	0	30
DstHostSerrorRate	0	1
NumRoot	0	993
Counted	0	511
DstBytes	0	5,155,468
SrcBytes	**0**	**693,375,640**
SrvCount	0	511
NumFailedLogins	0	5
NumFileCreations	0	28
Duration	0	58,329

Doherty et al (2007) have conducted experiments which show that normalisation of data values for a dataset may eliminate this problem. For this reason, the numeric-valued features of the KDD Cup 1999 dataset were normalised in the pre-processing step for the KNN algorithm. Secondly, the normalised values were mapped into the range [0, 1000] to avoid the effects of rounding when fractional values are multiplied together.

4.5 Sampling methods

All the experiments reported in this thesis involved the use of simple random sampling. Simple random sampling is the process of selecting a sample of the population units while giving every member of the population an equal chance of being selected (Rao, 2000). Simple random sampling may be done with replacement (SRSWR) or without replacement (SRSWOR). For SRSWOR, every population unit gets only one chance of being considered for selection. For SRSWR, every population unit gets many chances of being considered for selection. Sequential random sampling, described in the next section, was used to implement both SRWR and SRWOR for the large datasets used in the experiments. Sequential random sampling is discussed in section 4.5.1. The shuffling of data prior to sampling is discussed in section 4.5.2.

4.5.1 Sequential random sampling

Olken and Rotem (1995) and Olken (1993) have studied the use of sequential random sampling from databases. For sequential random sampling, the problem is to draw a random sample of size *n* without replacement, from a file containing *N* records. The simplest sequential random sampling method is due to Fan et al (1962) and Jones (1962), and proceeds as follows: An independent uniform random variate from the uniform interval [0,1] is generated for each record in the file to determine whether the record should be included in the sample. If n_t records have already been chosen from among the first *t* records in the file, the $(t+1)^{st}$ record is chosen with probability (RQ_{size}/RM_{size}), where $RQ_{size} = (n - n_t)$ is the number of records that still need to be chosen for the sample, and $RM_{size} = (N - t)$ is the number of records in the file still to be processed. Olken (1993) has used these methods to study database sampling.

4.5.2 Obtaining random samples from datasets

The records for each of the large datasets used in the experiments were randomised (shuffled) prior to sampling. The reason for shuffling the data was to remove any possible ordering in the dataset and to maximise the randomness of the order in which the instances appear. Sequential random sampling was then used to achieve simple random sampling, either from the whole dataset or from partitions of the dataset. In order to create bootstrap samples, the sequential random sampling procedure was repeated several times on the dataset, with a different random seed for each iteration. The shuffling and sampling from the datasets were implemented using stored procedures in a Microsoft SQL Server database.

4.6 The data mining algorithms used in the experiments

The two classification algorithms used for the experiments are decision tree for classification and K-Nearest Neighbour (KNN) classification. Decision tree classification (Quinlan, 1993; Quinlan, 1986; Breiman et al, 1984), which constructs classification models, has the desirable property that it attempts to identify the most relevant features. The KNN algorithm (Cover & Hart, 1967) is very different from the decision tree algorithms, as it does not perform feature selection of any kind, and therefore benefits the most from feature selection. Wu et al (2008) have reported that the decision tree algorithms as implemented in C4.5 (Quinlan, 1993; Quinlan, 1986) and CART (Breiman et al, 1984) as well as the KNN classification algorithm (Cover & Hart, 1967), are among the top ten algorithms used in data mining research. This section provides a brief description of the classification tree and KNN classification algorithms. Classification tree algorithms are discussed in section 4.6.1. KNN classification is discussed in section 4.6.2.

4.6.1 Classification trees

A classification tree algorithm creates a tree-structured model for the prediction of a qualitative variable called the class variable. In the classification tree model each leaf node provides information about the class to be assigned to instances that fall in that node. A classification tree algorithm recursively partitions a set of training data, using one predictive feature at a time, to create training dataset partitions. A classification tree is constructed, along with the partitioning process, based on the generated training dataset partitions. The heuristic used to guide the partitioning process uses a *class impurity* measure. At each decision point (for partitioning), all remaining features are evaluated. The feature that produces the partitions with the lowest *class impurity* is selected for partitioning. The selected feature then becomes the test for the decision/classification tree node with its values labeling the branches of the node. Commonly used class impurity measures are the chi-square (CHAID) criterion (Giudici, 2003), the two-ing criterion (Breiman et al, 1984), the Gini index of diversity (Breiman et al, 1984), and the entropy function (Quinlan, 1986).

The partitioning process should ideally stop when each partition is pure, that is, it consists of training instances all of the same class. In practice, however, pruning methods are used to halt the partitioning when it is no longer statistically valid to continue (Quinlan, 1993; Breiman et al, 1984). Breiman et al (1984) have observed that the tree growing procedures result in trees that are much larger than the data warrant. For example, if splitting is carried out to the point where each terminal node contains only one data case, then each node is classified by the case it contains, and the error on the training data is zero. This is an extreme case of overfitting which was discussed in chapter 2. On the other hand, when a tree is too small, then useful classification information in the training data has been ignored. This results in a high rate of classification error on the training data and a high predictive error rate on future instances. To determine the optimally-sized tree, a three step procedure is used (Osei-Bryson, 2004; Breiman et al, 1984). The first step is to grow a tree that is as large as possible. In the second step, the tree is pruned upward from the leaf nodes until the root is reached. In the third step, an independent sample of test data is used to estimate the predictive accuracy of all the pruned trees. The tree with the highest accuracy on the test data is selected as the optimally-sized tree. Optimisation methods from Operations Research have also been proposed for the selection of the optimal-size classification tree based on multiple objectives (Osei-Bryson, 2004). For the final classification tree that is used for classification, each leaf node has an assigned posterior probability for each class. In the prediction process, when a query instance lands at a given leaf node, the class with the highest probability at that node is predicted for the query instance (Osei-Bryson, 2004; Quinlan, 2004).

4.6.2 K-Nearest Neighbour classification

The K-Nearest Neighbour (KNN) classification algorithm originates from the field of statistical pattern recognition (Cover & Hart, 1967). The *inductive bias* of the KNN algorithm corresponds to an assumption that the classification of an instance $\mathbf{x}_q$, will be most similar to the classification of other instances that are nearby in terms of Euclidean distance. K-nearest neighbour classification uses a lazy algorithm which only constructs a classifier in the form of a target function, only when a new instance for classification is presented. The target function may be either discrete or real valued. If the target function is discrete valued then it is of the form $f : R^d \to C$, where *d* is the number of predictive features and *C* is the finite set $\{c_1,\ldots,c_k\}$ of the classes in the training data. For the simplest implementation of the KNN algorithm, the target function is estimated by computing a score for each class and returning that class that most frequently occurs among the K-nearest instances, based on the Euclidean distance. The score computed by the KNN algorithm is also the posterior probability $P_r(c_i \mid \mathbf{x}_q)$ that the query instance $\mathbf{x}_q$ belongs to class c_i. The computation of the Euclidean distance between query (test) instance $\mathbf{x}_q$ and training instance ***x*** is

$$dist(\boldsymbol{x}, \boldsymbol{x_q}) = \left(\sum_{i=1}^{d} (x_i - x_{qi})^2 \right)^{\frac{1}{2}} \qquad (4.1)$$

where *d* is the number of predictive features for the instance space. Since many datasets have qualitative features, special treatment is needed for the qualitative nominal and qualitative ordinal values when computing Euclidean distance. A common approach is to define the quantity $(x_i - x_{qi})$ for qualitative (nominal and ordinal) values as follows (Mitchell, 1997):

$$(x_i - x_{qi}) = \begin{cases} \textit{0 if the qualitative values are identical} \\ \textit{1 if the qualitative values are different} \end{cases} \qquad (4.2)$$

4.7 Measures of model performance

Evaluation is a crucial part of design science research. The measures for evaluating predictive model performance are discussed in section 4.7.1. Statistical methods for comparing two models on performance are discussed in section 4.7.2. ROC and lift chart analysis are discussed in section 4.7.3.

4.7.1 Measures of predictive performance

It was stated in chapter 1 that statisticians have invented effective methods of model construction, validation and testing for small datasets. Model validation and testing, using small amounts of data, has typically been done in the past using cross validation, the hold out method, or the bootstrap method (Mitchell, 1997; Hand, 1997; Moore & Lee, 1994). These methods were discussed in chapter 2. Even though the predictive performance of a model is very commonly reported in terms of predictive accuracy, especially in machine learning literature, various measures exist for more detailed analysis of the predictive capabilities of a model (Giudici, 2003; Hand et al, 2001; Hand, 1997). By generating a confusion matrix, performance measures can be computed for a given classification model. Table 4.9 shows a theoretical confusion matrix for a 2-class problem with two class labels *positive* and *negative* (Giudici, 2003; Hand, 1997). For a given validation dataset or test dataset, the value *TP* represents the number of positive instances that are correctly predicted as positive instances. The value *FN* represents the number of positive instances that are incorrectly predicted as negative instances. The value *FP* represents the number of negative instances that are incorrectly predicted as positive instances. The value *TN* represents the number of negative instances that are correctly predicted as negative instances. In general, a confusion matrix can be created for any *k*-class (*k* > *1*) classification model. From the *TP, FN, FP* and *TN* values, various performance measures can be derived (Giudici, 2003; Kubat & Matwin, 1997; Hand, 1997).

Table 4.10 gives the definitions and computation of the performance measures for a 2-class model. These performance measures provide useful information which can be used to compare classification models and to select the model that has the best predictive performance on the test data. The counts *FN* and *FP* represent levels of *class confusion*. The value *FN* represents the level to which instances of the *positive* class are mis-classified as *negative* instances and *FP* represents the level to which instances of the *negative* class are mis-classified by the model as positive instances.

Table 4.9: Theoretical confusion matrix for a 2-class model

Actual class	**Predicted class**		**Totals**
	positive	**negative**	
positive	*TP*	*FN*	*Pos = TP + FN*
negative	*FP*	*TN*	*Neg = FP + TN*
Totals	*TP + FP*	*FN + TN*	***Pos + Neg***

Table 4.10: Measures of performance derived from a confusion matrix

Measure			**Computation (in terms of table 4.9)**
Name	**Description**	**symbol**	
Error	error rate	*error*	*(FN + FP) / (Pos + Neg)*
Accuracy	Accuracy	*accuracy*	*(TP + TN) / (Pos + Neg)*
Sensitivity	True positive rate	*TPRATE*	*TP / (TP + FN)*
Specificity	True negative rate	*TNRATE*	*TN / (FP + TN)*
Precision	Correct positive prediction rate	*Precision*	*TP / (TP + FP)*
Type I error rate	False negative rate	*FNRATE*	*FN / (TP + FN)*
Type II error rate	False positive rate	*FPRATE*	*FP / (FP + TN)*
Y rate	Positive prediction rate	*YRATE*	*(TP + FP) / (Pos + Neg)*

The concepts of *positive instances* and *negative instances* for *k*-class prediction tasks were interpreted as follows in this thesis. Each class c_i was treated as the positive class in contrast to all the other *k-1* classes which were treated as the negative classes. This resulted in the creation and analysis of *k* confusion matrices with one 2 x 2 confusion matrix for each (positive) class.

The *error* and *accuracy* measures have a straight forward interpretation. In this thesis, the accuracy (rather than the *error*) is reported for all experiments on predictive performance. For a *2*-class problem, the *sensitivity* or *true positive rate* is the error rate on the test instances that belong to the *positive* class. For *2*-class problems, the *specificity* or *true negative rate* is the error rate on the test instances that belong to the *negative* class. The *false negative rate (type I error rate)* is the rate at which a model fails to classify *positive* instances as *positive*. The *false positive rate* (*type II error rate)* is the rate at which a model fails to classify *negative* instances as *negative*. The *YRATE* is used for lift analysis as discussed in section 4.7.3.

4.7.2 Statistical test to compare model performance

For purposes of comparing the performance of two predictive models M_A and M_B, a common approach is to establish the performance of each model on several test problems and compute the values of selected measures, or, all of the measures presented in the last section. Most commonly, in machine learning, the predictive accuracy or error rate are computed. Statistical tests are then used to compare the values of the measures on the test problems in order to establish if one model provides a higher level of predictive performance. Suppose that models M_A and M_B are each tested on a set of *n* problems, $PSet_A = \{problem_{A1},..., problem_{An}\}$ for model M_A and $PSet_B = \{problem_{B1},..., problem_{Bn}\}$ for model M_B. For statistical testing, when the sample size *n* is large (*n* ≥ *30*), the *Z* test for normal distributions is used to compare the mean values of the performance measures. When *n* is small (*5*≤ *n*<*30*), Student's t test is used to compare the mean values (Cohen, 1995).

There are two types of t-tests for comparing sample means. For the first t-test, called the *two sample t test* (Cohen, 1995), the problem sets $PSet_A$ and $PSet_B$ are different. For the second t-test, called the *paired sample t test* (Cohen, 1995), the problem sets $PSet_A$ and $PSet_B$ are identical. The models M_A and M_B are each tested on each problem, $problem_{Ai}$, and the statistical test is based on the difference in performance on each of the test problems. The *paired sample t-test* has more statistical power than the *two sample t-test* since it controls for (minimises) the variance due to the test problems (Cohen, 1995). The *paired sample t-test* was used for all the experiments in this thesis to compare model performance. In order to establish whether model M_A provides a higher level of predictive performance compared to model M_B, the following null hypothesis H_0 and *two-tail* alternative hypothesis H_a were tested:

$$H_0 : \mu_\delta = 0, H_a : \mu_\delta \neq 0 \tag{4.3}$$

where μ_A and μ_B represent the hypothesised mean values for one of the performance measures presented in the last section and $\mu_\delta = \mu_A - \mu_B$ represents the mean difference.

The F-test for variances (Cohen, 1995) was used in this thesis to compare the single and aggregate models in terms of variability of predictive performance. Cohen (1995: pg 205) has advised that comparison of performance variance for two models can be used to establish whether one model exhibits more erratic (or more coherent) behaviour compared with the other model. When two models have equal mean predictive performance then the model with more coherent performance should be preferred (Cohen, 1995). For the F-test of variances, the null hypothesis H_0 is that there is no significant difference in the performance variances of both models.

When an experiment is conducted, the probability of obtaining a particular sample result given the null hypothesis H_0 is called the *p value*. There are two methods of conducting statistical inference with p values. With the first, more traditional method, the researcher decides on the level of significance at which the null hypothesis will be rejected. Conventionally, a significance level of 0.05 (p = 0.05) is used. If the *p* value for a test is less than 0.05, then the null hypothesis is rejected (Montgomery et al, 2004; Cohen, 1995). For the second method, various levels of the *p* value are used to determine the outcome of the test, as shown in table 4.11 (Stirling, 2008). The second method of interpreting *p* values was adopted for the experiments of this thesis.

Table 4.11: Interpretation of p values for statistical tests

p value	**Interpretation**
$p < 0.01$	Strong evidence for the rejection of H_0
$0.01 < p \leq 0.05$	Moderate evidence for the rejection of H_0
$0.05 < p \leq 0.1$	Marginal or weak evidence for the rejection of H_0
$p > 0.1$	No evidence to support the rejection of H_0

4.7.3 Analysis of performance using ROC curves and lift charts

Receiver Operating Characteristic (ROC) curves and lift charts are commonly used as graphic representations of predictive model performance for 2-class prediction tasks (Giudici & Figini, 2009; Witten & Frank, 2005; Giudici, 2003; Berry & Linoff, 2000). A probabilistic classification model will typically assign a class and a score for the class. Most commonly, the score is the posterior probability that a test instance belongs to the predicted class (Giudici & Figini, 2009; Witten & Frank, 2005; Giudici, 2003; Berry & Linoff, 2000). ROC analysis and lift analysis are concerned with the selection of the model with the optimal performance based on the cut-off point λ that is used to decide when an instance should be declared positive or negative. A cut-off point is the score value $conf(\mathbf{x})$ for which $conf(\mathbf{x}) \geq \lambda$ implies that the predicted class for instance $\mathbf{x}$ is the positive class. ROC and lift analysis can also be used to determine which of two models provides a higher level of predictive performance as discussed below.

The Receiver Operating Characteristic (ROC) curve construct originates from signal detection applications where there is a signal transmitter and a signal receiver for a given (possibly noisy) transmission channel. A ROC curve is used to specify the relationship between the hit rate (correct detection) and the miss rate (false alarm rate) for the signal receiver (Witten & Frank, 2005). For classification modeling, a ROC curve is created using the information in a 2-class confusion matrix. More precisely, a ROC curve is a plot on a 2-dimensional Cartesian plane with the *x* and *y* values defined as (Vuk & Curk, 2006; Fawcett, 2001, 2004, 2006; Ferri et al, 2003; Hand & Till, 2001):

$$x = FPRATE(\lambda),\ y = TPRATE(\lambda) \qquad (4.4)$$

where $FPRATE(\lambda)$ and $TPRATE(\lambda)$ are respectively the false positive and true positive rates obtained when the cut-off value of λ is used. In order to understand the purpose of ROC analysis for classification modeling, it is useful to make a distinction between a *discrete classifier* and a *probabilistic classifier*. A discrete classifier assigns a class label to a test (or query) instance for a fixed threshold value (Fawcett, 2001, 2004, 2006). A probabilistic classifier on the other hand has the ability to assign a class label and a (probabilistic) score to a test (or query) instance for different threshold values. Stated differently, a probabilistic classifier operates in ROC space (Fawcett, 2001, 2004, 2006) which is the 2-dimensional plane defined by equation (4.4). A discrete classifier corresponds to exactly one point in the ROC space of a probabilistic classifier.

Figure 4.5 shows the Cartesian plane for the ROC space with a ROC curve example. A ROC curve represents relative tradeoffs between the benefits (true positives) and the costs (false positives) of using a given probabilistic classifier (Fawcett, 2006).

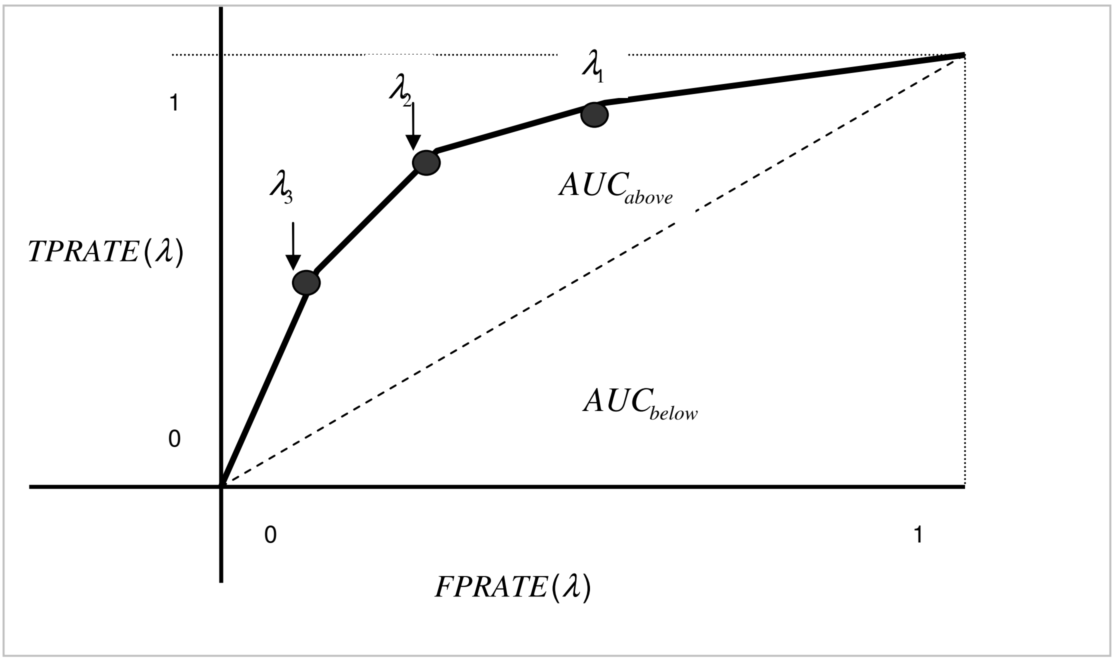

Figure 4.5: ROC space and AUC

For a given probabilistic classifier, each cut-off value of λ corresponds to a single point in the ROC space as defined in equation (4.4). The ROC curve joins these points for $-\infty < \lambda < \infty$. The point (0,0) represents a classifier which never gives a positive prediction. The point (1,1) represents a classifier which always gives a positive prediction. The point (0,1) represents a perfect classifier which never issues incorrect predictions. For the ROC curve example shown in figure 4.5, the relationship between the cut-off values is: $\lambda_3 > \lambda_2 > \lambda_1$, that is, the higher the cut-off value, the lower the FPRATE and TPRATE. A 45 degree line is normally plotted on the ROC plane to represent the ROC curve for classification in the absence of a model (random guessing). Any ROC point which lies below the 45 degree line represents a model which performs worse than random guessing.

An important statistic provided by the ROC curve is the Area Under the ROC curve (AUC). The AUC is the area between the x-axis, y-axis and the ROC curve (Fawcett 2001, 2004, 2006; Vuk & Curk, 2006; Ferri et al, 2003). This is the sum of the areas labelled AUC_{above} and AUC_{below} in figure 4.5. The area AUC_{below} has a fixed value of 0.5. Fawcett (2006) has observed that the area AUC_{above} is related to the Gini concentration coefficient (Breiman et al, 1984) as:

$$Gini = 2\text{x}AUC_{above} \tag{4.5}$$

Hand and Till (2001) have observed that the total area under the curve is related to the Gini concentration coefficient as:

$$Gini + 1 = 2\text{x}(AUC_{below} + AUC_{above}) \tag{4.6}$$

The definition of the Gini concentration coefficient is given in appendix B. The AUC is equivalent to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance (Fawcett, 2006). The AUC is also equivalent to the statistical Wilcoxon test of ranks (Fawcett, 2006; Hand & Till, 2001; Hanley & McNeil, 1982). Given two classifiers, the classifier with the larger AUC value provides a higher level of predictive performance. When the ROC curves of the two classifiers lie above the 45 degree line the performance difference is determined by the AUC_{above} area. For this reason, all discussions of the AUC provided in chapter 9 refer to the AUC_{above} area.

Two-class ROC analysis is concerned with the computation of the AUC, which is computed in a straight-forward manner by calculating the area under the ROC curve in the 2-diminsional Cartesian plane defined by equation (4.4). For *k*-class (*k* > 2) prediction tasks, ROC analysis is concerned with the computation of the Volume Under the ROC Surface (VUS). Computation and visualisation of the VUS is a non-trivial task. Two surrogate measures for the VUS, which have been proposed by Hand and Till (2001) and Provost and Domingos (2001) are discussed in chapter 9 of this thesis. The ROC (VUS) analysis results for the models studied in the experiments for this thesis are also presented in chapter 9.

The lift chart construct originates from the domain of predictive modeling for marketing and sales. For purposes of targeting customers in Marketing, the *lift factor* represents the expected increase in response rates when a model is used compared to the situation when no model is used to determine the customers to be targeted (Witten & Frank, 2005; Berry & Linoff, 2000). In order to plot a lift chart, the scores (probability values) assigned by the model on the test data are sorted into ascending (or descending) order and then grouped into deciles. A score for each group (decile) is then computed as the mean score within each group (Giudici & Figini, 2009; Witten & Frank, 2005; Giudici, 2003; Berry & Linoff, 2000). More precisely, a lift chart is a plot on a 2-dimensional Cartesian plane with the *x* and *y* values defined as (Vuk & Curk, 2006):

$$x = YRATE(\lambda),\ y = TPRATE(\lambda) \tag{4.7}$$

The lift factor for each decile is computed as the ratio between the score assigned by the model and the score when no model is used (random guessing). The lift chart is plotted with the deciles on the horizontal axis and the cumulative lift factor values on the vertical axis. A baseline line that represents random guessing is also plotted. As for ROC curves, the area between the base line and the cumulative lift curve indicates the quality of the model. The larger the area, the better the model. A discussion of why lift analysis was not used is provided in chapter 6.

4.8 Software used for the experiments

Various software packages were used for the experiments as shown in table 4.12. The datasets were stored in a Microsoft SQL Server database. Storing the datasets in a database made it especially easy to establish the composition of each dataset, and to pre-process the KDD Cup 1999 dataset, using SQL statements and stored procedures. Dataset sampling was implemented using stored procedures implemented in the Microsoft SQL Server procedural language.

Table 4.12: Software used for the experiments

Task / Activity	Software
Dataset storage and retrieval	MS SQL Server 2000
Dataset sampling	Stored procedures implemented in the MS SQL Server procedural language
Measurement of correlation coefficients	Specialised code implemented in Borland C++ Builder 5
Feature subset search	Specialised code implemented in Borland C++ Builder 5
KNN classification (modeling)	Specialised code implemented in Borland C++ Builder 5
Classification tree modeling	See 5 – Windows version of the C5.0 classifier
Aggregate modeling	Specialised code implemented in Borland C++ Builder 5
ROC analysis	Specialised code implemented in Borland C++ Builder 5
Statistical hypothesis testing	SPSS versions 15 and 17
Generation of descriptive statistics for datasets	SPSS versions 15 and 17, MS SQL Server 2000 SQL, Ms Excel 2003
Various activities	MS Excel 2003

The See5 classifier (Quinlan, 2004), which is the MS Windows version of the C5.0 classifier for Unix, was used for classification tree construction. SPSS versions 15 and 17 for MS Windows were used for conducting the Student's t-tests for the statistical analysis of model performance. Specialised applications were created in Borland C++ Builder 4 and 5 for measuring class-feature and feature-feature correlation coefficients, feature selection, the KNN classifiers, aggregate model classifiers, and ROC analysis. It should be pointed out that statistical software provides functions for correlation measurement and ROC analysis. However specialised software was implemented in order to speed up the experiments.

4.9 Chapter summary

The main research questions, central argument of this thesis, and research methods have been presented and justified in this chapter. The design science research paradigm in conjunction with the scientific method were used as a conceptual framework for the research. The datasets used for the experiments were obtained from the UCI KDD Archive and UCI Machine Learning repository. The descriptive statistics of the datasets, as well as the pre-processing that was done on the datasets have been discussed. Sequential random sampling was used to obtain random samples from large datasets. The algorithms that were used for modeling, namely: classification tree and K-Nearest Neighbour, have been presented. The measures of predictive performance that were used in the experiments have been discussed. Finally, the software used for the experiments has been presented. In the next four chapters, the experiments that were conducted, the results that were obtained and the proposed methods for feature and dataset selection are presented. The theoretical models that were deduced from the experimental results are discussed in chapter 10.