Spring 2014

Backward Sequential Feature Elimination And Joining Algorithms In Machine Learning

Sanya Valsan
San Jose State University

Backward Sequential Feature Elimination And Joining Algorithms In Machine Learning

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Sanya Valsan

May 2014

The Designated Project Committee Approves the Project Titled

Backward Sequential Feature Elimination and Joining Algorithms
In Machine Learning
by

Sanya Valsan

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2014

Dr. Sami Khuri Date

Department of Computer Science

Dr. Chris Pollett Date

Department of Computer Science

Dr. Chris Tseng Date

Department of Computer Science

ABSTRACT

Backward Sequential Feature Elimination and Joining Algorithms In Machine Learning

By Sanya Valsan

The Naïve Bayes Model is a special case of Bayesian networks with strong independence assumptions. It is typically used for classification problems. The Naïve Bayes model is trained using the given data to estimate the parameters necessary for classification. This model of classification is very popular since it is simple yet efficient and accurate.

While the Naïve Bayes model is considered accurate on most of the problem instances, there is a set of problems for which the Naïve Bayes does not give accurate results when compared to other classifiers such as the decision tree algorithms. One reason for it could be the strong independence assumption of the Naïve Bayes model. This project aims at searching for dependencies between the features and studying the consequences of applying these dependencies in classifying instances. We propose two different algorithms, the Backward Sequential Joining and the Backward Sequential Elimination that can be applied in order to improve the accuracy of the Naïve Bayes model. We then compare the accuracies of the different algorithms and derive conclusion based on the results.

ACKNOWLEDGEMENTS

I am extremely obliged to my project advisor, Dr. Sami Khuri, for his guidance, encouragement, and support throughout this project. I would also like to thank my committee members, Dr. Chris Pollett and Dr. Chris Tseng for their time and support.

My special thanks to Natalia Khuri and Professor Fernando Lobo for their invaluable insights and suggestions during the course of this project.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

In the real world, in order to perform tasks we need to reason by obtaining information and draw conclusions based on the information collected. Consider a doctor who takes information from a patient in order to diagnose a disease the patient may be suffering from. He may take information such as the patient's symptoms, test results, personal characteristics such as the height, weight and so on. We can develop a computer program for this particular domain and train the system to make predictions based on the patient's answers. However, by doing so, the flexibility of the system is reduced. If there is a case where we need to change the questions or the domain, then the answers will not be found in the system that we train. In other words, the system becomes too rigid. Hence, we will first have to bring about major changes in the system itself.

A different approach to solving the above problem is to use an approach called *declarative representation*. We can develop a model which understands how the system works. This model can then be applied to answer questions pertaining to various categories or domains. It makes the system more flexible. This model can be developed using various algorithms, declarative representation being one of them. As Daphne Koller, a professor at Stanford University claims, "The key property of a declarative representation is the separation of knowledge and reasoning. The representation has its own clear semantics, separate from the algorithms that one can apply to it. Thus, we

can develop a general set of algorithms that apply to any model within a broad class, whether in the domain of medical diagnosis or speech recognition. Conversely, we can improve our model for a specific application domain without having to modify our reasoning algorithms constantly."[1] In this project, we focus on models which hold a certain degree of uncertainty. We study and implement the methods by which we can improve the accuracy of the probabilistic models.

Uncertainty arises due to the limitations in one's potential to understand and decipher the true state of any given system. These limitations could be due to the partial information that one has access to, noisy observations and so on. Thus, in order to draw substantial conclusions, one needs to reason not only the possibilities but also the probabilities.

1.2 Probabilistic Models

A model is a declarative representation of how we understand the real world. It deals with how different objects interact with each other. These models represent complex systems. The complex systems are characterized by the presence of different features, which may or may not be interrelated. These features are called random variables which have values depending on how the system has been described. For example, a person believed to have tuberculosis will have cough as one of his symptoms. Cough is thus a random variable with two values *present* or *absent*. In order to reason these using probabilistic principles, one needs to construct a joint distribution over a set of random variables. By doing so, one will be able to answer an extensive range of intriguing questions.

1.3 Probabilistic Graphical Models

A probabilistic graphical model provides a mechanism for exploiting the structure in complex distributions to describe them in a compact way using a graph which represents the conditional dependence structure between random variables. It is a framework which deals with the uncertainty involved with modelling applications which have a large number of parameters or variables. Figure 1 shows a word cloud representing the most commonly used terms with respect to Probabilistic Graphical models.

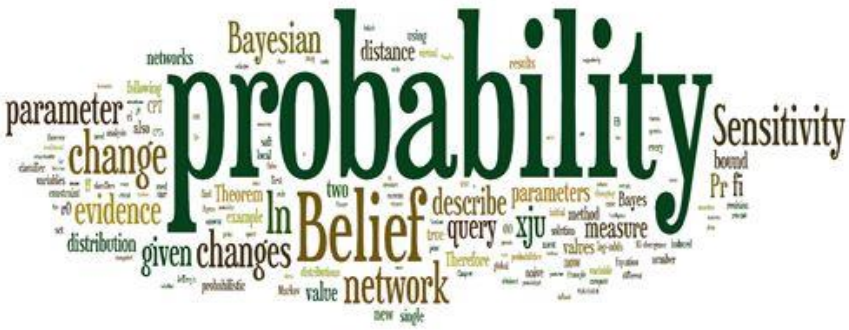

Figure 1: Word cloud representing Probabilistic Graphical Models [2]

Thus, Probabilistic graphical models are a classic schema which integrates probabilities and independence constraints to depict complex, real world outcomes. They can help us predict systems and draw inferences based on the information provided. They can be described as a generalized form of many known models such as the Hidden Markov models.

1.4 Overview

The schema of probabilistic graphical models is quite vast, and involves a variety of models. In this report, we study the classification algorithms based on Bayesian networks.

In Chapter 2, we describe the important concepts related to probability theory. In Chapter 3, we present background information about the Bayesian network representation which is based on directed graphs. Chapter 4 illustrates the Naïve Bayes Model and its application in classification problems. In Chapter 5, we compute the accuracies for the Naïve Bayes classifier for different databases and propose algorithms which search for dependencies amongst the attributes in order to improve the Naïve Bayes accuracy. Chapter 6 covers the results for the three algorithms implemented in this project namely, Naïve Bayes classifier, Backward Sequential Elimination and Backward Sequential Joining and draws conclusions.

CHAPTER 2

Foundations

In this chapter, we will learn some important concepts pertaining to the probability theory.

2.1 Probability

Probability is a measure of the likelihood that an event occurs. It is defined as a ratio of number of favourable outcomes to the total number of possible outcomes in an experiment.

Example 1.1: If you flip a coin, the probability of the result being a 'head' is $\frac{1}{2}$, since the coin has two sides. If we roll a 6-side die, the probability of one occurring is $\frac{1}{6}$.

Each of the results in the above example can be described as an event. We measure the probability of the event occurring which is also known as *Prior Probability*.

2.2 Probability Distributions

A probability distribution is a mapping of all the possible values of a random variable to their corresponding probabilities for a given sample space. It can be written as $P(X = x)\ or\ P(x)$. [3]

2.3 Conditional Probability

Conditional Probability is the probability that an event will occur given that another event has already occurred. In other words, consider that we have two events A and B. Conditional Probability is the probability that the event B occurs given the

information that the event A has occurred. This can be written as $P(B|A)$ which denotes probability of B given A. This is also known as Posterior Probability.[4]

When two events are not independent, the probability of both occurring is given by,

$$P(A\ and\ B) = P(A)\ P(B|A) \qquad \text{Eq. 2.1}$$

Hence,

$$P(B|A) = \frac{P(A\ and\ B)}{P(A)} \qquad \text{Eq. 2.2}$$

However, when the two events are independent, the probability of B does not depend on event A occurring. In this case,

$$P(B|A) = P(B) \qquad \text{Eq. 2.3}$$

Example 2.1: Let us take an example of a card game, wherein a player has to draw out two cards belonging to the same type if he wants to win the game. A card stack has a total of 52 cards. There are 13 cards of the same type and there are four types of cards in all. Suppose the player first picks out a spade. Now, there are 12 spades remaining in the stack of 51 cards. The player would like the next card he picks to be a spade.

We can write the conditional probability as:

$$P(Draws\ a\ second\ spade \mid First\ drawn\ is\ a\ spade) = \frac{12}{51}$$

This can be interpreted as the probability that the second card drawn is a spade given that the first card drawn was a spade.

2.3.1 Conditional Probability for Multiple Events

Conditional Probability can also be expressed as a combination of multiple events. Let $A_1, A_2 \ldots A_K$ be mutually exclusive and exhaustive events. Then, for any other event B,[5]

$$P(B) = P(B|A_1) \,.\, P(A_1) + P(B|A_2) \,.\, P(A_2) + \ldots + P(B|A_k)|\, A_k$$

$$= \sum_j P(B|A_j)P(A_j) \qquad \text{Eq. 2.4}$$

Thus,

$$P(A_i|B) = \frac{\mathrm{P}(B|A_i)\mathrm{P}(A_i)}{\sum_j P(B|A_j)P(A_j)} \qquad \text{Eq. 2.5}$$

2.4 Random Variables

A random variable is a variable whose possible values are numerical outcomes of a random phenomenon.[6]

Example 2.2 [7]*:* Consider a student who is intelligent and his intelligence is described by a variable *'I'* which can have values *high* and *low*. The student is taking a class. The difficulty of the class can be represented by a variable *'D'* which can be *difficult* or *not difficult*. The third variable is the grade *'G'* which can have three values *A*, *B* and *C*. The student's SAT scores is represented by *'S'* and can have values as either *low* or *high*. Also, the letter of recommendation represented by *L* can have values *good* or *not good*. Thus, in the student example *I,D,G,S,L* are *random variables* (see Figure 2).

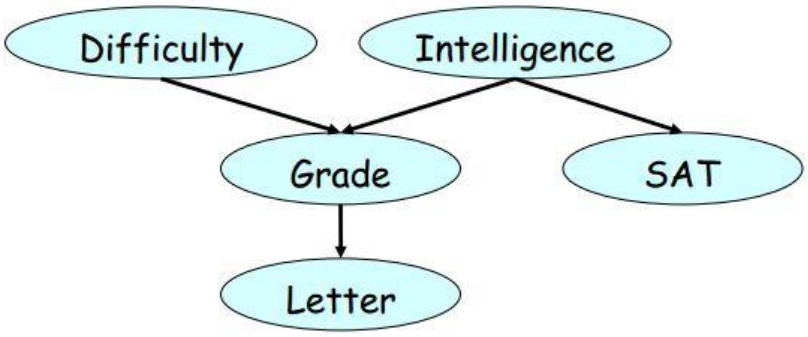

Figure 2: The Student Example [7]

Random variables can be discrete or continuous.

Discrete Random Variables

A discrete random variable is a variable which takes only a finite number of distinct values[6].For example, in Figure 2, *I* (Intelligence) can take only two distinct values *low* and *high*. Similarly, *G* can take only three values *A*,*B*,*C*. Thus, these variables are discrete random variables.

Continuous Random Variables

A continuous random variable is a variable which takes an infinite number of possible values[6]. It is defined over an interval or range of values. For example, the weights of a group of people can be continuous values ranging from 60 to 100 pounds.

2.5 Marginal and Joint distribution

Marginal distribution is the distribution over all the events that can be described in terms of the random variable. For example, in Figure 2, *P(I=high)* and *P(I=low)* are specific events over which the marginal distribution can be computed for the variable intelligence.

One may also be interested in the values of several random variables. For example, one might be interested in the event *"Intelligence = low"* and *"Grade=A"*. In this case, we need to compute the joint distribution over these two random variables. Thus, the joint distribution over a set $X = \{X_1, \dots, X_N\}$ is represented by $P(X_1, \dots, X_N)$ and is a distribution that assigns probabilities to events that can be described in terms of the random variables $X_1, \dots, X_N$.

In the next chapter, we describe the Bayesian network representation (a probabilistic graphical model) which is based on directed acyclic graphs.

CHAPTER 3

Background

3.1 Representation

Probabilistic Graphical Models can be represented in different ways. Two popular ways of representing them is by using Bayesian Networks and Markov Models. Bayesian networks are a graph-based representation of a set of random variables and their conditional dependencies. It is very useful in a variety of applications such as bioinformatics, gene expression, information retrieval, semantic search, image processing and many other applications.

3.2 Bayesian Network Fundamentals

A Bayesian network is represented by a directed acyclic graph (DAG). Acyclic means that it has no cycles, i.e. one cannot reverse the edges and get back to where one started. It is abbreviated as a DAG and denoted by the letter *R* in this report. This graph R can be viewed either as a data structure that represents the joint distributions or a representation for a set of conditional independence assumptions. The student example, in Figure 2, is a Bayesian network represented as a directed graph where the nodes represent the random variables and the edges represent direct influence between the variables.

In the student example, in Figure 2, we have a student who's taking a class for a grade. If the student is intelligent, then the grade of the student will be good. Therefore, the letter of recommendation will also be good. However, if the student is not intelligent then the student will not receive a good grade. Hence, the letter of recommendation will not be good. The grade of the student also depends on the difficulty of the class. If the class

was difficult the student will receive a low grade. However, if the class is easy the student will receive a good grade and hence a good letter of recommendation.

Thus, the course difficulty (D) and the intelligence (I) of the student are independent variables and the student's grade (G) depends on these two factors. The student's SAT score(S) depends only on his intelligence and the letter of recommendation (L) depends on the student's grade in the class. The student example represents a joint probability distribution via the chain rule for Bayesian networks.[8]

The rule is written as:

$$P(D, I, G, S, L) = P(D)\ P(I)\ P(G|I, D)\ P(S|I)\ P(L|G) \qquad \text{Eq. 3.1}$$

In the Bayesian network for the student example in Figure 2, the nodes of the directed acyclic graph represents random variables from X_1 to X_N. For each node in the graph, X_i, we have a Conditional Probability Distribution *(CPD)* that denotes the dependence of X_i on its parents in the graph *R*. This would be the probability of *G* given *I* and *D* written as P (G|I,D). Here, X_i would be *G* and its parents would be *I* and *D*.

Definition of Bayes Chain Rule:

Let R be a Bayesian Network graph over the variables X_1 ,…,X_N,.The distribution P over the same space factorizes according to R if P can be expressed as a product of the conditional probabilities.[8]

$$P(X_{1\ldots}\ X_N) = \sum_{i=1}^{n} P(X_i|Pa(X_i)) \qquad \text{Eq. 3.2}$$

The equation is called as chain rule for Bayesian networks. The individual factors $P(X_i|Pa(X_i))$ are called conditional probability distributions (CPDs). Here, $Pa(X_i)$ represents the parent(s) of the variable X_i in the network.

For developing a Bayesian network, initially we construct a DAG and then the CPDs for each random variable are calculated given its parents in the DAG. If the product of these CPDs gives the joint distribution, then the graph is defined to be a Bayesian network.

3.3 Reasoning Patterns

Now that we have defined Bayesian networks, we study some of the reasoning patterns that allow models to perform. When we condition on certain variables, it affects the joint probability distributions. Some reasoning patterns are observed based on how it affects the probabilities.

3.3.1 Causal Reasoning

Consider the example shown in Figure 3(a). If intelligence is low, the probability of getting a good letter of recommendation goes low. But if intelligence is low and the class is also difficult, the grade and hence the probability of getting a good letter of recommendation increases. Cases such as these, where there is a top to bottom influence of various factors, illustrate causal reasoning.

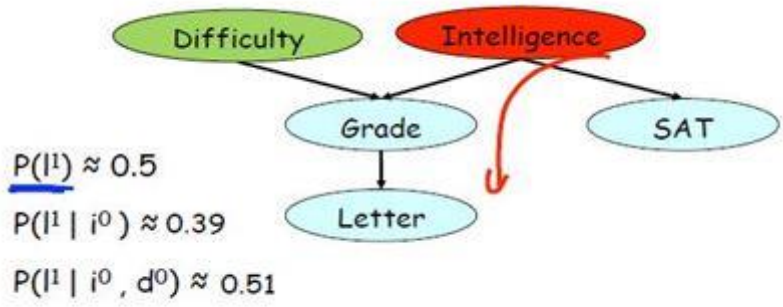

Figure 3(a): Causal Reasoning Example [9]

3.3.2 Evidential Reasoning

Consider the example shown in Figure 3(b). If the student gets a low grade, the probability of the class being difficult increases. Also, the probability of the student being intelligent becomes low if he gets a low grade. Cases such as these, where there is a bottom up influence of various factors, illustrate evidential reasoning.

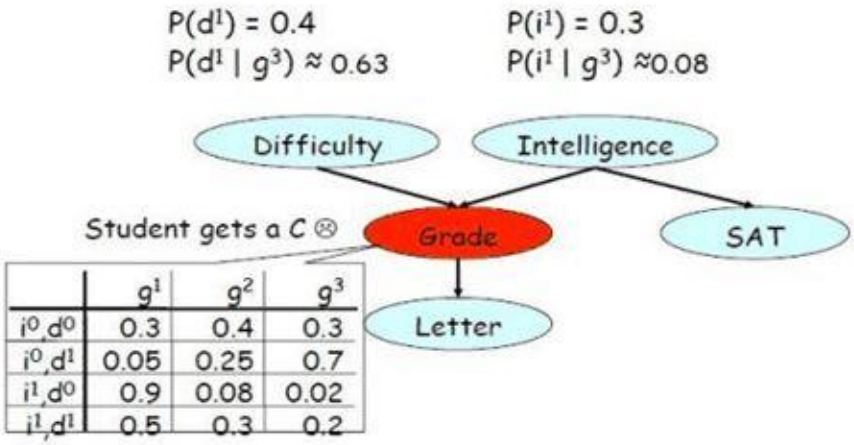

Figure 3(b): Evidential Reasoning Example [9]

3.3.3 Intercausal Reasoning

This reasoning depends on both the parent and child. As shown in Figure 3(c), given the condition that the grade is low, the probability of the student being intelligent decreases. However, given the condition that the grade is low and the class is difficult, the probability of the student being intelligent increases.

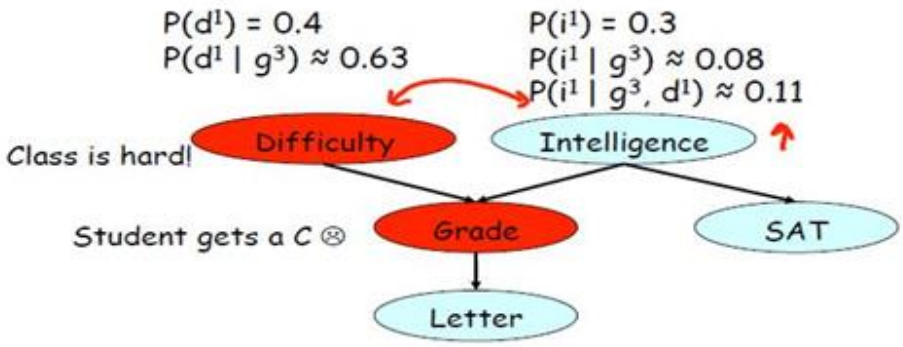

Figure 3(c): Intercausal Reasoning [9]

3.4 Independencies in Bayesian Networks

Bayesian networks have independence assumptions which state that the variables are independent of each other. Also, there is a relationship between factorization of distribution as a product of factors and the independence assumption. If we have, $P(X,Y)$ as the product of $P(X)$ and $P(Y)$, then $P(X)$ and $P(Y)$ are independent of each other.

In the next chapter, we study the Naïve Bayes model in detail.

CHAPTER 4

The Naïve Bayes Model

4.1 Introduction

The Naïve Bayes model is a special case of the Bayesian networks which holds strong independence assumption. Such naïve assumption not only reduces the complexity of the model but surprisingly gives accurate results. It is typically used for classification where there exists a set of features for a set of instances and we have to determine to which class a given instance belongs.

Bayes' theorem in probability statistics is a theorem on the probabilities of events A and B, $P(A)$ and $P(B)$ and the conditional probabilities of $P(A|B)$ and $P(B|A)$.

In Bayesian interpretation, probability measures the degree of belief. Therefore, for proposition *A* and evidence *B*,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \qquad \text{Eq. 4.1}$$

Where, $P(A)$ – the prior probability,

$P(A|B)$ – the posterior probability,

and the quotient $\frac{P(B|A)}{P(B)}$ is the dependency of *A* on *B*.

4.2 Naïve Bayes Classifier

The Naïve Bayes classifier is a probabilistic classifier based on applying Bayes theorem. It can predict class membership probabilities i.e. a Bayesian classifier predicts the probability that a given sample in the data belongs to a particular class. Given a sample X, the classifier will predict if X belongs to the class having highest posterior probability conditioned on X. In general, the working of the Naïve Bayes classifier is as follows:

Let *T* be a training set of samples, each having the class labels $C_1, C_2, \dots, C_k$. A sample X is represented as a set of *'n'* normally valued attributes $\{x_1, x_2, \dots, x_n\}$ having values A_1, $A_2, \dots, A_n$ resprectively. A sample X is predicted to belong to class C if and only if,[10]

$$P(C_i|X) > P\left(C_j\middle|X\right) \forall\, i \leq j \ \leq n\,, i \neq j \qquad \text{Eq. 4.2}$$

The class C_i for which $P(C_i|X)$ is maximized is called the maximum posterior hypothesis.

Now we have, by Bayes theorem, as seen in Equation 4.1 [10]

$$P(C_i|X) = \frac{P(X|C_i)\, P(C_i)}{P(X)} \qquad \text{Eq. 4.3}$$

As $P(X)$ is the same for all classes, only $P(X|C_i)P(C_i)$ needs to be maximized.

$P(X|C_i)$ can be computed from the data.

$P(C_i)$ the prior probability can be calculated by [10],

$$P(C_i) = \frac{freq(C_i, T)}{|T|} \quad \text{Eq. 4.4}$$

The computation time of $P(X|C_i)$ increases in cases where the given data set has a large number of attributes. So in order to reduce the computation time, the naïve assumption that the values of the attributes are conditionally independent of each other is made. Hence, the classifier gets the name Naïve Bayes classifier.

With the conditional independence assumption,[10]

$$P(X|C_i) \approx \prod_{k=1}^{n} P(x_k|\, C_i) \quad \text{Eq. 4.5}$$

$P(x_1 = A_1|\, C_1)$, $P(x_2 = A_2|\, C_2) \ldots P(x_n = A_n|\, C_i)$ can be estimated from the training set.

4.3 Types of Attribute Data

4.3.1 Categorical Attributes

If the data in the sample is categorical, then $P(x_k|C_i)$ is the number of instances of class C_i in the training set having the value x_k for attribute A_k divided by number of times the class C_i appears in the training set.

4.3.2 Continuous Attributes

There are two ways to deal with continuous attributes:

1) The data are divided into their categorical counterparts. This process is known as discretization or binning. Binning improves accuracy of the predictive models by reducing the noise or non-linearity. They are of two types [11].

a) Unsupervised

It converts continuous data into its categorical counterparts by either Equal Width or Equal Frequency. They do not depend on class information.

Equal Width Binning

The algorithm divides the data into k intervals of equal size. The width of the intervals is:

$$w(i) = \frac{maximum\ value\ of\ range\ - minimum\ value\ of\ range}{k} \qquad \text{Eq. 4.6}$$

The interval's boundaries now become,

$$min + w(i), \min + \big(2 \times w(i)\big) \ldots min + \big((k-1) \times\ w(i)\big).$$

We put the continuous data value for each attribute according to the interval they belong to.

Example 4.1: Consider we have data containing ages of different individuals. In order to discretize it using equal width binning, we compute the following.

Data : 0,5,6,10,17,19,23,27,30,31,32,35,39

To classify it into five different intervals, i.e. k=5

Interval 1: [0, 7)

Interval 2: [8, 15)

Interval 3: [16, 23)

Interval 4: [24, 31)

Interval 5: [32, 39)

Equal Frequency Binning

The algorithm divides the data into k groups which have approximately the same count of values. Then each value of the attribute is observed to see which group it belongs to and is placed accordingly.

Example 4.2: Using the data in example 4.1, we perform computations using the equal frequency binning.

Data: 0,5,6,10,17,19,23,27,30,31,32,35,39,39

Group 1: [0, 10)

Group 2: [11, 20)

Group 3: [21, 30)

Group 4: [31, 40)

b) Supervised:

Supervised Binning makes use of the class information when selecting discretization cut points. Entropy based binning is an example of supervised binning.

2) The other method to calculate the probability distribution for continuous variables is to calculate the normal distributions for numerical variables. The probability density function for normal distribution is defined by two parameters namely, mean and standard deviation.[11]

Mean (μ) [11]

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{Eq. 4.7}$$

Standard Deviation (σ) [11]

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2\right]^{0.5} \qquad \text{Eq. 4.8}$$

Normal Distribution $(f(x))$ [11]

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \qquad \text{Eq. 4.9}$$

Next, we consider an example to calculate the probability distributions using probability density functions.[11]

Example 4.3: Suppose we have the weather data (humidity) which will help us to decide if a particular day, is a good day to play tennis. The data is shown in Table 1[11]. We calculate the probabilities for Play Tennis = Yes and Play Tennis = No.

Table 1: Example showing μ and σ for given values of humidity [11]

Play Tennis	Class	Humidity Data (%)	Mean	Standard Deviation
	Yes	86, 96, 80, 65, 70, 80, 90, 75	79.1	10.2
	No	85, 90, 70, 95, 91	86.2	9.7

Using the mean and standard deviation, we can calculate the normal distribution for both values for playing tennis.

P (humidity = 74 | play = yes) = 0.0344

P (humidity = 74 | play = no) = 0.0187

Thus, it can be concluded from the probabilities that it is indeed a good day to play tennis.

4.4 Using the Naïve Bayes Classifier

Next, we study an example to predict the class of a given instance with the Naïve Bayes approach.[4]

Example 4.4: The iHealth Database

There is a company called iHealth which sells two models of wearable exercise monitors. The two models are called i100 and i500. One has to build a recommendation system which will help them sell the right model to the customer. For this, the customer first fills out a questionnaire. It consists of questions which relate to the attributes of the model. An example of the questionnaire results is shown in Table 2[4]. Here, the model

type is a class (in this example i500 and i100 are the two classes) and Main Interest, Current Exercise, Motivation, Comfort level are the attributes for this training set.

Table 2: Training data from the iHealth database [4]

Main Interest	Current Exercise Level	How motivated	Comfort with tech. devices	Model #
Both	Sedentary	Moderate	Yes	i100
Both	Sedentary	Moderate	No	i100
Health	Sedentary	Moderate	Yes	i500
Appearance	Active	Moderate	Yes	i500
Appearance	Moderate	Aggressive	Yes	i500
Appearance	Moderate	Aggressive	No	i100
Health	Moderate	Aggressive	No	i500
Both	Active	Moderate	Yes	i100
Both	Moderate	Aggressive	Yes	i500
Appearance	Active	Aggressive	Yes	i500
Both	Active	Aggressive	No	i500
Health	Active	Moderate	No	i500
Health	Sedentary	Aggressive	Yes	i500
Appearance	Active	Moderate	No	i100
Health	Sedentary	Moderate	No	i100

Now consider a person’s answers has the following attributes

Instance

Main Interest: health

Current exercise level: moderate

Motivation: moderate

Comfortable with technological devices: yes

Classification

Using Naïve Bayes, one has to calculate the probabilities,

P (i100| health, moderate, moderate, yes) and

P (i500|health, moderate, moderate, yes).

The class which has a higher probability will be the most suitable model for the customer.

For this, one has to compute the probabilities for each of the attribute values for a given class.

Table 3: Probability of the attribute '*Main Interest*' for the given class

Main Interest	**i500**	**i100**
Both	1/6	1/3
Health	2/6	2/3
Appearance	3/6	0/3

Table 4: Probability of the attribute '*Exercise Level*' for the given class

Exercise Level	i500	i100
Sedentary	2/9	3/6
Moderate	3/9	1/6
Active	4/9	2/6

Table 5: Probability of the attribute '*Motivation*' for the given class

Motivation	i500	i100
Moderate	3/9	5/6
Aggressive	6/9	1/6

Table 6: Probability of the attribute '*Comfort Level*' for the given class

Comfort Level	i500	i100
Yes	6/9	2/6
No	3/9	4/6

Now, the probabilities for each class for the given instance are calculated:

P (i500|health, moderate, moderate, yes)

= (4/9 ˣ 3/9 ˣ 3/9 ˣ 6/9) ˣ 9/15 = 0.01975

P (i100| health, moderate, moderate, yes)

= (1/6 ˣ 1/6 ˣ 5/6 ˣ 2/6) ˣ 6/15 = 0.00309

If the probabilities for the two classes are compared, it can be seen that i500 has a higher probability. Hence, the customer should be offered the model i500 since it will be best suited for him.

4.5 Laplacian Correction

When we calculate the probabilities, there can be cases where we get a zero probability. This happens when none of the samples of that class has a given attribute value. Consider there was a class C_i, and X had an attribute value x_k, which is not seen in any of the instances of class C_i for that attribute. Thus, using Equation 4.5,

$$P(x_k|C_i) = 0,$$

and when this is multiplied with the probabilities of all other attributes we still get zero probability. In such cases, the Laplacian correction is used.

Assume that the training set is large enough that even if one is added to each count it would make a very negligible difference in the estimated probabilities, but at the same time it will help to overcome the zero probability value problems. If there is *p* counts to which one is added each, then p counts should be added to the corresponding denominator used in the probability calculation.

Consider the following example.

Example 4.5: In Example 4.4, consider that the dataset contained ten samples, and there are zero instances with interest equal to moderate, five instances with interest equal to appearance and five instances with interest equal to both. The probabilities of these events, without Laplacian correction, are 0, 0.5(5/10) and 0.5(5/10) respectively. Now, Laplacian correction is used where it is assumed that there is one more sample

for each interest-value pair. In this way, the following corrected probabilities are obtained:

1/11 = 0.091 6/11=0.545 6/11= 0.545

The corrected probability estimates are close to their uncorrected counterparts, yet the zero probability value is avoided.

4.6 Popularity

The Naïve Bayes classifier is very popular because it involves very basic mathematical calculations. Classifying becomes easy and efficient. Bayes classifier has worked very well in many complex real-world situations. On many problems, the accuracy of Naïve Bayes is equal to or higher than many machine learning algorithms.

In the next chapter, we discuss the methods of finding dependencies between the attributes of the Naïve Bayes classifier.

CHAPTER 5

Searching for Dependencies

The Naïve Bayes classifier has strong independence assumptions. It assumes that the presence or absence of a feature is completely independent of the presence or absence of any other feature.

5.1 Conditional Independence

Consider a general probability distribution $P(x_1, x_2)$ of two variables $x_1 \; and \; x_2$ [12]

Using Bayes rule as seen in Equation 4.3,

$$P(x_1, x_2) = P(x_1|x_2)\, P(x_2) \qquad \text{Eq. 5.1}$$

Now, consider there is another class variable *'c'*, which can be written as,

$$P(x_1, x_2 \mid c) = P(x_1|x_2, c)\, P(x_2|c) \qquad \text{Eq. 5.2}$$

If the information provided about c is sufficient to determine how x_1 will be distributed, then there is no need to know the information about x_2.

Thus Equation 5.2 can be re-written as,

$$P(x_1|x_2, c) = P(x_1|c) \qquad \text{Eq. 5.3}$$

To give a generalized example of conditional independence, consider the following example,[12]

Example 5.1:

$$P(\text{cloudy},\text{windy}|\text{storm}) = P(\text{cloudy}|\text{windy},\text{storm})\ P(\text{windy}|\text{storm})$$

Now, if we consider

$$P(\text{cloudy}|\text{windy},\text{storm}) = P(\text{cloudy}|\text{storm})$$

the distribution becomes,

$$P(\text{cloudy},\text{windy}|\text{storm}) = P(\text{cloudy}|\text{storm})P(\text{windy}|\text{storm})$$

5.2 The Problem

Michael Pazzani, a professor at University of California, Irvine conducted research on the accuracies given by different datasets using the Naïve Bayes classifier.[13] He tested the accuracy of the Naïve Bayes classifier for different datasets from the UCI repository. He observed that on many problems the accuracy of the Naïve Bayes classifier is equal to or greater than that of more sophisticated machine learning algorithms. He compared the results with another simple decision tree algorithm called ID3. On each problem, both algorithms were run 24 times on the same training set and tested on the same disjoint test sets. The accuracy was determined by calculating the proportion of agreements between predicted and actual classes. He found that on most of the problems, the Naïve Bayes classifier is more accurate than ID3. However, there is a set of problems for which the accuracy of Naïve Bayes classifier was significantly less accurate than the decision tree algorithm using the paired two-tailed

t-test.

According to Pazzani, one possible explanation for the accuracies to be significantly less for certain datasets is that the independence assumption of the Naïve Bayes does not hold in these cases. Recall that the independence assumption states that the attributes of the sample are independent of each other within a class.

The aim of this project is to search for dependencies among pairs of attributes and try to improve the accuracies of the databases in consideration. Datasets and databases will be used interchangeably in this report. In order to look for dependencies, feature selection algorithms can be applied to the classifier. A feature selection algorithm helps in determining new subsets of features along with evaluation measures which calculates the scores for the different feature subsets. The wrapper method is one category of the feature selection algorithm. Wrapper methods are predictive models which calculate and evaluate scores for different feature subsets. Each subset initially trains the model and then uses a test set to evaluate it. Determining the accuracy from the test set gives a score for that particular feature subset. These wrapper methods employ different operations which can be used to perform a feature selection[14]. Figure 4 shows the wrapper approach for the feature subset selection. In this figure, the induction algorithm is considered as a black box. The training set and test set are provided to the induction algorithm with different features removed from the dataset. The feature set with the highest evaluation is chosen to be run by the induction algorithm and the final accuracy is estimated.

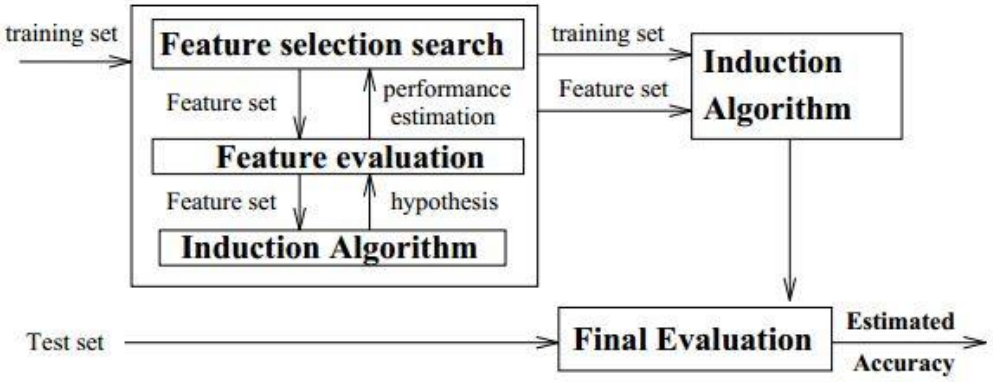

Figure 4: A wrapper approach to feature subset selection [15]

Pazzani proposed two operations in order to carry out feature subset selections namely, Joining of Attributes and Elimination of Attributes, which will be discussed in detail in the next section.

5.3 Joining of Attributes

Joining is an operation that creates a new compound attribute which replaces the original two attributes that were used for joining. The values of the new attributes are the combinations of the values of the original attributes. This operation is also known as constructive induction. “Constructive induction is the process of changing the representation of examples by creating new attributes from existing attributes.”[16]

In this project, the joining of the values of the attributes is done by taking the Cartesian product of the two values. For example, if there were two attributes height and age, the new attribute formed height_age will have values tall_old, tall_young, short_old and short_young. If the value of either attribute is unknown, then the value of the joined attribute also cannot be determined. Cartesian products will be explained in further detail later in this chapter. Joining is possible only on discrete values. When we deal with continuous valued attributes, we first need to convert the continuous data to

discrete data using one of the binning methods described in the Chapter 4. For the purpose of this project, we have used the equal width binning method. Attribute joining results in an evitable elimination step. When two attributes are joined, the original attributes are eliminated. The original attributes cannot be joined with other attributes again. Due to this, the dimensionality of the dataset is reduced by one for every joining that takes place. By repeated application of the joining operator, more than two attributes may be joined. Joining attributes corresponds to the creation of a “hidden” variable as shown in the Figure 5.

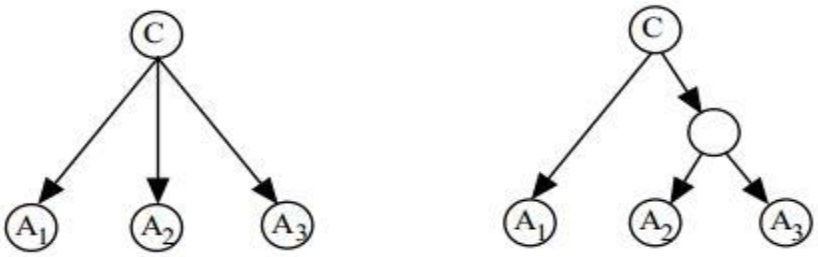

Figure 5: Joining attributes introduces a hidden variable in the Bayesian network [16]

Joining attributes has one potential limitation. When attributes are joined, there will be relatively less data to compute the joint probabilities from the data. This may sometimes results in inaccurate results. One way of dealing with this problem is to join only those attributes from which the accuracy estimates show improvement. This approach has been adopted during the implementation of this project.

5.4 Elimination of Attributes

Elimination is the process of attribute reduction where the attributes are deleted in turn and the accuracy of the classifier is determined from the remaining set of attributes. The elimination operation is used to see if certain attributes contribute more information

towards the probability estimations than other attributes. If the accuracy improves by deleting certain attributes, then a new classifier is returned containing all but the deleted attributes. This process is repeated till there is no further improvement in the accuracies.

Cartesian product

The Cartesian product is used for attributes is used for joining attributes. Pazzani(1998) proposed the Cartesian product operation for joining attributes. The Cartesian product joins two attributes A_1 and A_2 into a joined attribute $A_1.A_2$, taking x values in the Cartesian product[17].

$$\{<a_i, a_j> \mid a_i \in Values(A_1) \hat{} \, a_j \in Values(A_2)\} \quad \text{Eq. 5.4}$$

where $Values(A_1)$ is the value set of attribute A.

There are advantages of using Cartesian product. In the Naïve Bayes classifier, attributes are treated individually and the product of individual probabilities calculated from the training data gives the joint probability. When the attributes are joined using Cartesian product, the joint probabilities can be calculated in an equivalent manner.

5.5 Finding Dependencies between Attributes

Pazzani(1998) suggested an algorithm known as Backward Sequential Elimination and Joining Algorithm to improve the accuracy of sample sets where the accuracy of the Naïve Bayes classifier was significantly less than the accuracy given by other decision tree algorithms. Pazzani, compared the results of this algorithm with the results of a simple decision tree algorithm using a paired two tailed t-test. He found that for a set of problems the Bayesian classifier is significantly less accurate than the decision tree at

the .05 level. In order to find ways to improve the accuracies on the given set of problems under consideration, he proposed the Backward Sequential Elimination and Joining Algorithm. This algorithm violates the independence assumption of the Naïve Bayes and searches for dependencies between the attributes.

5.5.1 Backward Sequential and Elimination Algorithm (BSEJ)[13]

The algorithm initializes a set of attributes which is used by the classifier. This set of attributes does not contain the joined attributes but the original attributes that were present in the sample set. Next, two operators are used to generate new classifiers:

a) Consider joining each pair of attributes and replacing the original attributes that were used to join the attributes.

b) Consider deleting each attribute used by the classifier

For every step in the classifier, every joining of attributes is considered and evaluated. If there is no improvement in the accuracy, then the classifier is returned with no change in the representation. If there is an improvement due to the application of operations on the classifier then the change is retained and the new classifier is returned. The same procedure is performed till there is no improvement or all the attributes in the classifier are exhausted. Following successful joining, the Backward Sequential Elimination and Joining(BSEJ) algorithm carries out an explicit elimination step. It deletes every attribute in turn which includes the joined attributes looking for the classifier which returns the best accuracy.

According to Pazzani, this approach, brings about a significant difference in the accuracies of the datasets for which the Naïve Bayes model showed less accuracy. Thus, searching for dependencies among attributes causes an increase in the accuracy.

5.6 A Wrapper Approach for Creating Cartesian Product Attributes and Elimination

In this project, the accuracy obtained by the use of the constructive induction(joining) operation is contrasted with that obtained by the use of attribute elimination operation for each dataset. Thus, we determine the accuracies of the sample set considering only one operation at a time. Initially, the accuracies of the sample set are calculated using the joining operation. This algorithm is called as the Backward Sequential Joining(BSJ).[17] Next, we determine the accuracies of the sample set considering only the elimination operation. This algorithm is known as Backward Sequential Elimination(BSE)[17]. The average accuracy obtained by each of the algorithms is compared with the Naïve Bayes classifier.

5.7 Wrapper Approach for Backward Sequential Joining (BSJ)

5.7.1 Implementation Steps

1) Calculate the accuracy for the dataset using Naïve Bayes classifier.

2) Set the original accuracy to the accuracy computed in Step 1.

3) For every ordered combination of two attributes, compute the accuracy of the classifier and set it as the new accuracy.

4) Compare the new accuracy with the original accuracy.

5) If the new accuracy is higher than the original one, return the new classifier with the joined attributes, else return the classifier of Step 2.

6) Continue this process till there is no further improvement in the accuracies or all of the attributes have been exhausted.

The pseudo code in Figure 6 outlines the implementation of the BSJ algorithm. During each iteration, the algorithm calculates the resulting accuracy of the ordered combination. The combination that results in the maximum increase in accuracy is the final classifier.

A Wrapper Approach the for Backward Sequential Joining(BSJ) Algorithm

Algorithm 1: BSJ(T)

acc $\leftarrow$ Accuracy (T) for the current classifier

success $\leftarrow$ true

while (success) **do**

success $\leftarrow$ false

for every ordered combination of two attributes A_i and A_j in T **do**

Produce T' from T by joining A_i and A_j, putting them on the position

$k \in \{i, j\}$

newAcc $\leftarrow$ accuracy (T') for current classifier

if (newAcc ≥ acc)

then

acc $\leftarrow$ newAcc

winner $\leftarrow$ T'

success $\leftarrow$ true

if success == true

then

$T \leftarrow$ winner

return T

Figure 6: Pseudo Code : Backward Sequential Joining [17]

5.8 Wrapper Approach for Backward Sequential Elimination

5.8.1 Implementation Steps

1) Calculate the accuracy for the dataset using Naïve Bayes classifier.

2) Set the original accuracy to the accuracy computed in Step 1.

3) Compute the accuracy of the classifier by removing each attribute one at a time and set it as the new accuracy.

4) Compare the new accuracy with the original accuracy.

5) If the new accuracy is higher than the original one, return the new classifier with the new subset of attributes, else return the classifier of Step 2.

6) Continue this process till there is no further improvement in the accuracies or all of the attributes have been exhausted.

The pseudo code in Figure 7 outlines the implementation of the BSE algorithm. During every iteration, it calculates the effect of eliminating each attribute in turn, considering only those attributes which result in an increase in accuracy.

A Wrapper Approach for the Backward Sequential Elimination(BSE) Algorithm

Algorithm 2: BSE(T)

acc $\leftarrow$ Accuracy (T) for the current classifier

success $\leftarrow$ true

while (success) **do**

success $\leftarrow$ false

for every attribute A in T **do**

Produce T' by removing A from every instance in T

newAcc $\leftarrow$ accuracy (T') for current classifier

if (newAcc ≥ acc)

then

acc $\leftarrow$ newAcc

winner $\leftarrow T'$

success $\leftarrow$ true

if success == true

then

$T \leftarrow$ winner

return T

Figure 7: Pseudo Code: Backward Sequential Elimination [17]

5.9 Experiments and Validation

In order to determine if there is an improvement in the accuracies, experiments were conducted using different datasets for the two algorithms i.e. BSJ and BSE. The flowchart in Figure 8 depicts the workflow of the application. The application reads the input file and determines if the data are discrete or continuous valued. If the data are continuous, they are converted into its discrete counterparts using equal width binning. Next, the prior probabilities, conditional probabilities and posterior probabilities are calculated and the accuracy of the classifer is determined. This accuracy is then used as a threshold for the BSJ and BSE algorithms. Then, depending on the algorithm selected by the user, the BSJ or BSE classifier is executed.

Ten-fold cross validations were performed on each dataset and the mean accuracy of the ten folds was computed. The highest accuracy obtained was taken as the final result. We chose this validation method because it is the fastest and most efficient way to determine the accuracy. Leave one out cross validation is not performed because it has high computation cost as our sample set is very large.

When there is no increase in the accuracy, the classifier is not modified to adjust the joined or deleted attributes. In other words, we do not change the representation of the sample set when there is no improvement in the accuracy by applying the changes. For example, if joining the attributes height and weight as height_weight does not increase the accuracy as compared to the original classifier, where the two attributes were not joined, then the original classifier is retained.

Whenever there is a good change in the representation of the sample set by joining or deletion of certain attributes, the change is retained. One aspect to be careful about is,

when new attributes are joined it replaces the attributes from which it was formed. This means that the attributes cannot be still retained in the new classifier in their original form. Also, it cannot be joined again to other attributes. For example, we cannot have height_weight and height_age as attributes in the same classifier. Once you have joined the attributes height_weight, the classifier will have height_weight and age as two different attributes. This is done since Bayesian classifiers assume independence of attributes within each class and the Cartesian product of two attributes is clearly dependent on the original attributes.

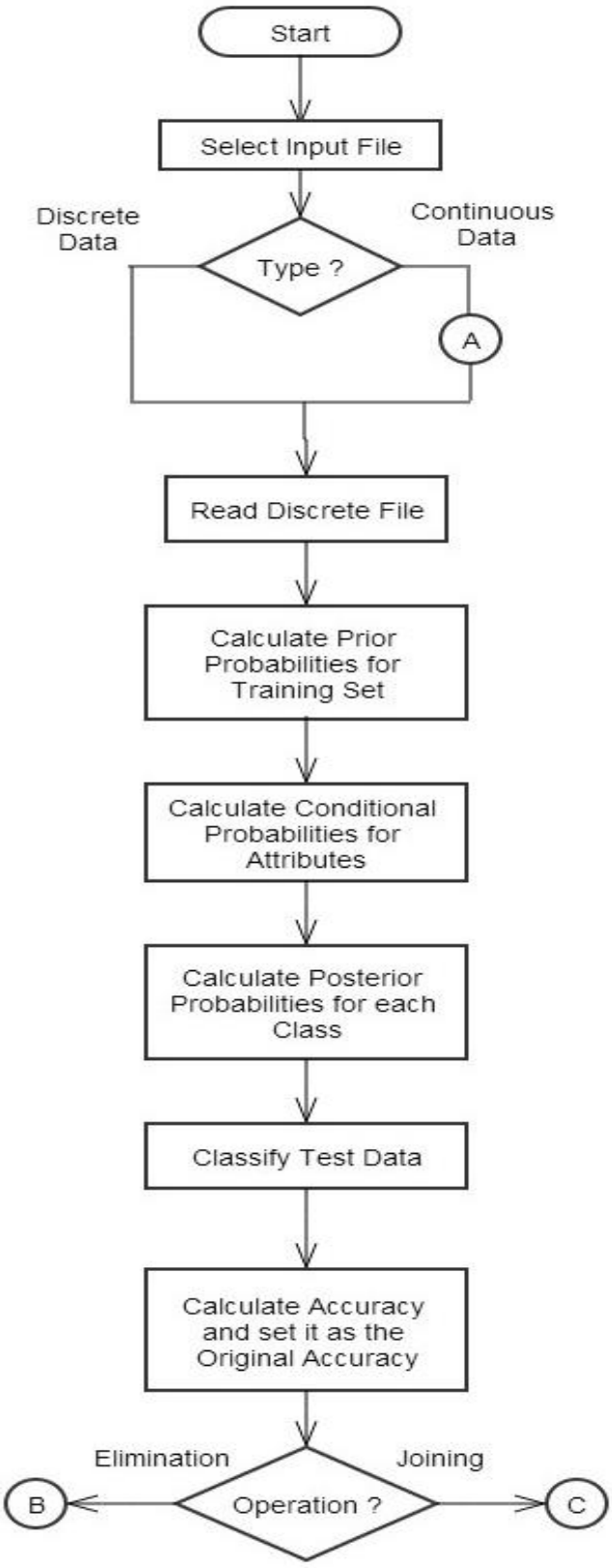

Figure 8: Flowchart for BSJ and BSE Algorithms

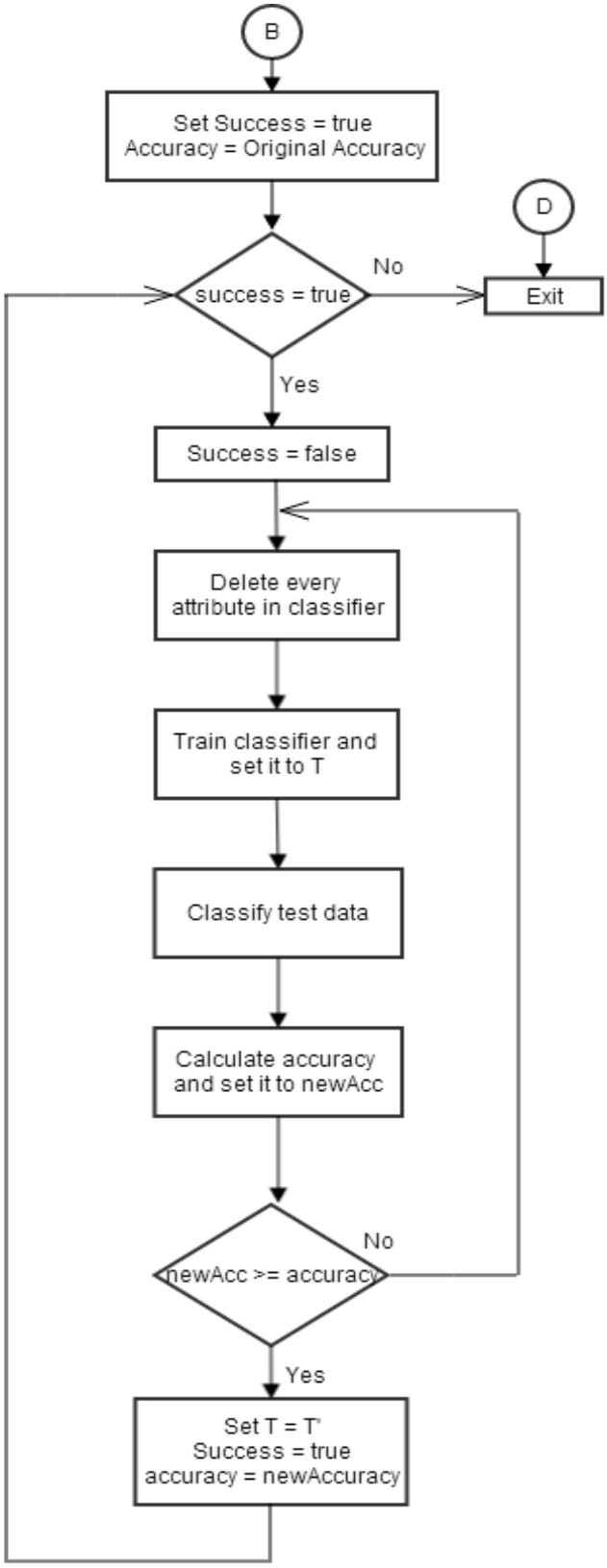

Figure 8: Flowchart for BSJ and BSE Algorithms(Continued)

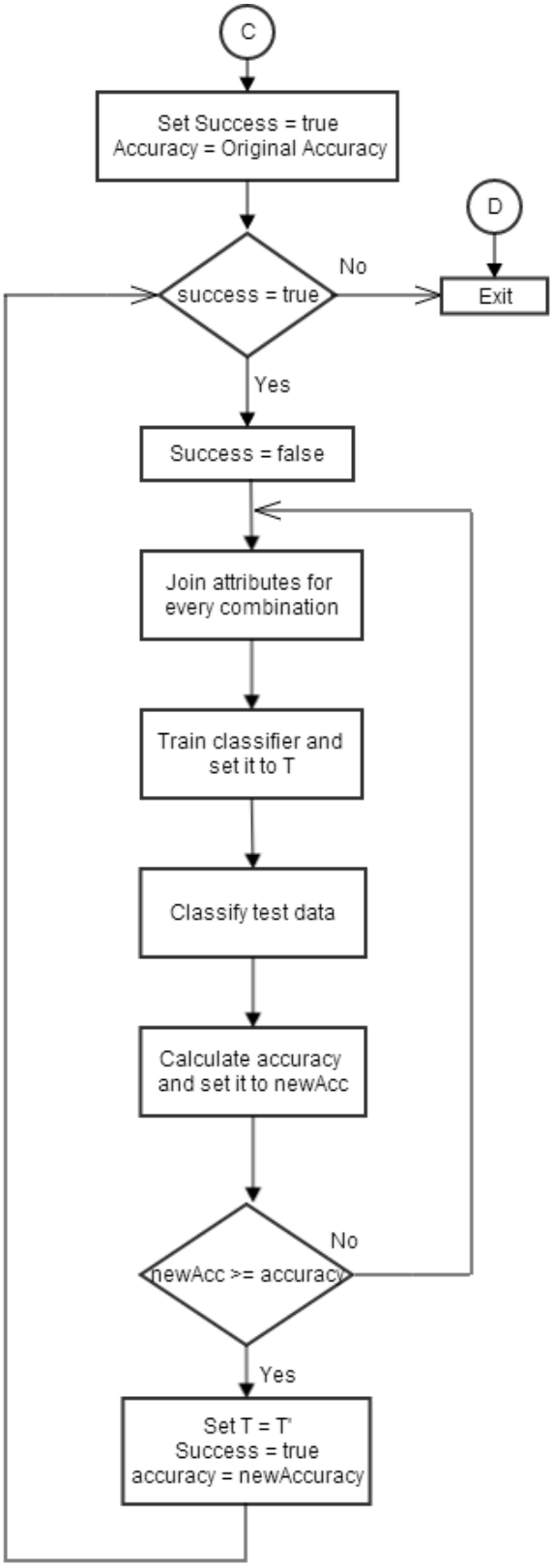

Figure 8: Flowchart for BSJ and BSE Algorithms(Continued)

In order to investigate the effects of the three learning algorithms namely Naïve Bayes, Backward Sequential Joining and Backward Sequential Elimination, ten databases from the UCI Repository of machine learning databases were used. In each experiment, there were ten trials of randomly selected training and test examples.

The next chapter illustrates the results (ten-fold cross validation) obtained by using the three algorithms on different datasets.

CHAPTER 6

Datasets and Results

6.1 Datasets

The datasets for this experiment were collected from the KEEL and UCI data repositories.

KEEL

KEEL is a data repository which contains a big collection of machine learning data which can be used for classification including standard, multi instance, regression and unsupervised learning.[19]

UCI Repository

The UCI Machine Repository is a collection of databases, domain theories and data generators that are used by researchers, students, educators and developers of the machine learning community. It was developed by two students from the University of California, Irvine. They currently maintain 284 machine learning datasets.[21]

6.2 Training Set and Test Set

All machine learning algorithms are trained for supervised learning, such as classification and prediction. We need to train them on particular inputs. Later, we can test them for inputs they have never seen before. They either classify the input or predict them based on their learning.

The input data initially needs to be divided into a development set and a test set. The development set consists of training set. The test set consists of an evaluation set. The

model has to learn from the training set and classify inputs that are not present in the training set.

The test set has data which are unknown and have never been seen before. However, the format of the test set should be similar to the training set. We must always ensure that the training set and test set are distinct otherwise the model would have already known the input and can give very high scores which could be misleading. If the test example is a subset of the training set, then we will always be close to 100% accurate which is overly optimistic. Usually, the training set will contain 80% of the original example. The rest can be used for the test set.

Ten-fold cross validation

Dividing the data set into two parts for training and testing respectively could lead to inaccurate classification. There can be a scenario where the training data set contains many examples having similar values for attributes and could get classified into the same class. Thus, the accuracy of the test set will be poor.

The ten-fold cross validation provides a way to solve this problem of inaccuracy. In this method, we divide the data into ten parts as seen in Figure 9. Nine parts are used for training and the rest one tenth is used for testing. Accuracy is determined for this set using the classifier. The process is then repeated for each of the one tenth parts. The average of all the accuracies then determines the accuracy of the classifier using ten-fold cross validation.

Cross validation can be n-fold, i.e. we can have one-fold, two-fold cross-validation.

Table 7: A simple example of a three-fold cross validation

Iteration 1	Train with parts 1 and 2	Test with part 3
Iteration 2	Train with parts 1 and 3	Test with part 2
Iteration 3	Train with parts 2 and 3	Test with part 1

The measures we obtain using ten-fold cross-validation are more accurate than the ones obtained by two-fold or three-fold validation because each time we train the classifier we are using 90% of the data compared to only 50% for two-fold cross-validation.

Leave-one-out cross validation

In this method of validation, we perform n-fold cross validation where n is the number of entries in the data set. For example, if there are 100 entries in a dataset , we train 99 entries and test the remaining one entry. We repeat this process singling out each entry of the dataset. In this way, the largest possible part of the original dataset is used for training. This results in a higher accuracy of the classifier. The advantage is that the results are deterministic. However, the computational time will be large.

6.3 Accuracy Test

The accuracy test used to determine the accuracy of the classifier in this project is the number of examples for which the instance were correctly classified divided by the total number of instances to be classified.[13]

6.4 Benchmark

In order to validate the results of the Naïve Bayes classifier implemented in this project, the results of the Naïve Bayes classifier as computed by the WEKA Tool[18][20] were used as a benchmark. WEKA is a classification package available online. Another

benchmark used is the results from Pazzani's paper on 'Searching for dependencies'.[13]

Datasets

Table 8 represents the number of features, classes, number of training and test instances of each dataset used for the ten-fold cross validation experiments.

Table 8: Information about the datasets used for the tenfold validations[19]

Sr. no	Dataset	No. of Features	Classes	# of Instances	Training Instances	Test Instances
1.	Glass	9	7	217	191	26
2.	Wine	13	3	178	160	18
3.	Iris	4	3	150	134	16
4.	Pima	8	2	768	690	78
5.	Hepatitis	19	2	155	139	16
6.	Yeast	8	10	1484	1335	149
7.	Wisconsin	9	2	683	630	53
8.	Tic-tac-toe	9	2	958	863	95
9.	Wdbc	30	2	569	512	57
10.	Letter	16	26	20000	17997	2003
11.	Heart	13	2	270	250	20
12.	Vowel	13	11	990	909	81

6.5 Comparative Analysis

6.5.1 Naïve Bayes Classifier Results

For the purpose of this project, the Naïve Bayes classifier was implemented and ten-fold cross validations were performed on different databases. As seen in Table 9, each row

in the table represents the results of one database. Each column represents the fold's result for the particular database. The last column in the table represents the mean (average of ten-folds) for each database. The results of the Naïve Bayes classifier are compared with the benchmarks namely the results of the Weka tool for the same datasets and the results presented by Pazzani[13]. For the domains namely glass, yeast and tic-tac-toe, the mean value of the accuracies are relatively smaller as compared to the other domains. One possible reason for this could be the naïve assumption of the independencies among the attributes.

6.5.2 Backward Sequential Joining and Backward Sequential Elimination Classifier Results

Tenfold cross validations were performed for the Backward Sequential Joining Algorithm(BSJ) and Backward Sequential Elimination(BSE) Algorithm implementation by using the same databases that were used for Naïve Bayes classification experiment. As seen in Table 10 and Table 11, each row in the table represents the results of one database. Each column represents the fold's result for the particular database. The last column in the table represents the mean (average of ten-folds) for each database. In both classifiers, the mean value of the accuracies for the domains, glass and tic-tac-toe improved considerably as compared to the Naïve Bayes classifier.

6.5.3 Comparative Analysis of the Algorithms

The results of the Naïve Bayes algorithm implementation are compared with the Backward Sequential Joining(BSJ) and Backward Sequential Elimination(BSE) algorithm implementation results.

As seen in Table 12, each row represents one of the databases used for the experiments. The second column represents the results obtained using the Weka Tool for Naïve Bayes classification. The third column represents the results obtained by Pazzani for his implementation of the Naïve Bayes classifier. The '–' symbol in certain cells represent the results which were not available for comparison. The fourth column shows the average of the ten-fold cross validation implemented in this project for Naïve Bayes classifier. The fifth column shows the average of the ten-fold cross validation for the implementation of the BSJ algorithm and the sixth column shows the average of the ten-fold cross validation for the implementation of the BSE algorithm.

Table 9: Ten-Fold Cross-Validation Results (Naïve Bayes Classifier)

Datasets	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Mean
Glass	47.83	52.17	69.56	36.36	63.63	50.00	33.33	65.00	50.00	61.11	52.90
Wine	88.88	94.44	100.0	100.0	94.12	100.0	100.0	100.0	100.0	88.88	96.63
Iris	93.33	73.33	100.0	86.66	86.66	93.33	93.33	100.0	93.33	100.0	92.00
Pima	70.13	73.07	81.58	68.83	77.92	72.37	77.92	81.58	76.62	71.43	75.15
Hepatitis	81.81	83.33	100.0	85.71	71.43	100.0	100.0	83.33	37.5	90.91	83.40
Yeast	55.03	51.67	48.99	55.03	56.08	48.65	45.94	50.67	48.65	47.29	50.80
Wisconsin	92.54	89.85	83.95	82.09	83.33	82.35	85.71	82.35	84.06	82.86	85.21
Tic-tac-Toe	71.57	65.26	65.62	69.79	66.66	72.92	70.83	67.71	65.62	78.12	69.41
Wdbc	89.47	80.70	89.47	87.71	82.45	68.42	80.70	85.96	75.43	71.42	81.17
Letter	72.79	73.45	73.95	74.03	74.15	73.23	71.84	73.68	74.78	74.61	74.96
Heart	77.78	92.59	77.78	85.19	85.19	92.59	88.89	81.48	66.67	85.19	83.33
Vowel	51.52	59.60	68.69	59.60	58.59	67.68	63.64	62.63	68.69	64.65	62.53

Table 10:Ten-Fold Cross-Validation Results (Backward Sequential Joining Algorithm)

Datasets	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Mean
Glass	60.89	60.86	69.56	59.09	72.72	72.72	71.42	80.00	60.00	72.22	67.95
Wine	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Iris	93.33	93.33	100.0	86.66	93.33	93.33	93.33	100.0	100.0	100.0	95.33
Pima	80.52	79.48	86.84	81.81	83.11	82.89	84.41	88.15	77.92	80.51	82.56
Hepatitis	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	83.33	100.0	98.33
Yeast	61.07	55.70	53.02	57.71	59.46	56.08	54.72	53.37	56.08	53.37	56.06
Wisconsin	98.50	97.10	98.55	100.0	98.48	98.52	100.0	100.0	97.10	95.71	98.39
Tic-tac-Toe	84.21	82.10	86.45	86.45	85.41	86.45	90.62	83.33	83.33	88.54	85.69
Wdbc	100.0	100.0	100.0	100.0	98.24	98.24	94.73	98.24	96.49	98.21	98.41
Letter	100.0	100.0	100.0	100.0	96.49	94.73	91.22	96.49	96.49	96.49	97.18
Heart	88.89	100.0	85.19	88.89	92.59	92.59	96.30	85.19	81.49	100.00	91.11
Vowel	86.87	82.83	92.93	79.80	85.86	82.83	86.87	86.87	85.86	85.86	85.66

Table 11: Ten-Fold Cross-Validation Results (Backward Sequential Elimination Algorithm)

Datasets	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Mean
Glass	60.87	56.52	69.56	54.54	68.18	72.72	42.85	70.00	50.00	61.11	60.64
Wine	100.0	100.0	100.0	100.00	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Iris	93.33	93.33	100.0	86.66	93.33	93.33	93.33	100.0	100.0	100.0	95.33
Pima	79.62	76.92	85.52	76.62	81.81	80.26	83.11	85.52	77.92	74.02	79.83
Hepatitis	100.0	100.0	100.0	100.00	85.71	100.0	100.0	83.33	87.5	100.0	95.65
Yeast	55.70	52.34	51.00	55.70	58.11	52.70	53.37	50.67	50.67	49.32	52.96
Wisconsin	97.01	97.10	98.55	100.00	100.0	97.05	100.0	100.0	97.10	94.28	98.11
Tic-tac-toe	71.57	65.26	65.62	69.79	66.66	72.91	70.83	67.70	65.65	78.12	69.41
Wdbc	100.0	100.0	100.0	100.00	96.49	94.73	91.22	96.49	96.49	96.49	97.18
Letter	74.13	74.55	75.35	74.73	75.4	74.73	73.14	74.68	75.63	76.11	74.85
Heart	85.19	96.30	92.59	88.89	92.59	92.59	96.30	85.19	77.78	100.00	90.74
Vowel	57.58	65.66	73.74	63.64	61.62	69.70	71.72	67.68	70.71	65.66	66.77

Table 12: A Comparison Ten-Fold Cross-Validation Accuracy Results

Dataset	Weka's Naïve Bayes	Naïve Bayes (Mean)	Backward Sequential Joining (Mean)	Backward Sequential Elimination (Mean)
Glass	69.99	52.90±15	67.95±8	60.64±12
Wine	–	96.63±8	100.0±0	100.0±0
Iris	94.00	92.00±6	95.33±9	95.33±9
Pima	77.86	75.15±7	82.56±7	79.83±6
Hepatitis	85.16	83.40±12	98.33±15	95.65±4
Yeast	–	50.80±6	56.06±5	52.96±6
Wisconsin	95.99	85.21±7	98.39±4	98.11±6
Tic-tac-toe	–	69.41±4	85.69±5	69.41±9
Wdbc	–	81.17±13	98.41±4	97.18±6
Letter	74.96	74.96±2	97.18±6	74.85±3
Heart	83.70	83.33±6	91.11±8	90.74±8
Vowel	63.53	62.53±5	85.66±6	66.77±7

As seen in Table 12, the mean values of the NB classifier and the mean values of the BSJ classifier for the ten databases were compared using the paired two-tailed T-test. Similarly, the mean values of the NB classifier and the mean values of the BSE were compared. Also, the mean values of the BSE and BSJ were compared using the same T-test.

The p-value (P) calculated for the algorithms using the paired two-tailed test are:

T-Test (NB and BSJ) - 0.0000645

T-Test (NB and BSE) - 0.001736

T-Test (BSJ and BSE) - 0.021902

If $P<0.001$, then there is very strong evidence against the null hypothesis in favor of the alternative[22]. This means that there is a difference between the accuracy of the two classifiers and BSJ performs better than NB from the observed accuracies.

If 0.001<P<0.01, then there is strong evidence against the null hypothesis in favor of the alternative[22]. This means that there is a difference between the accuracy of the two classifiers and BSE performs better than NB from the observed accuracies.

If 0.01<P<0.05, then there is moderate evidence against the null hypothesis in favor of the alternative[22]. This means that there is a difference between the accuracy of the two classifiers and BSJ performs better than BSE from the observed accuracies.

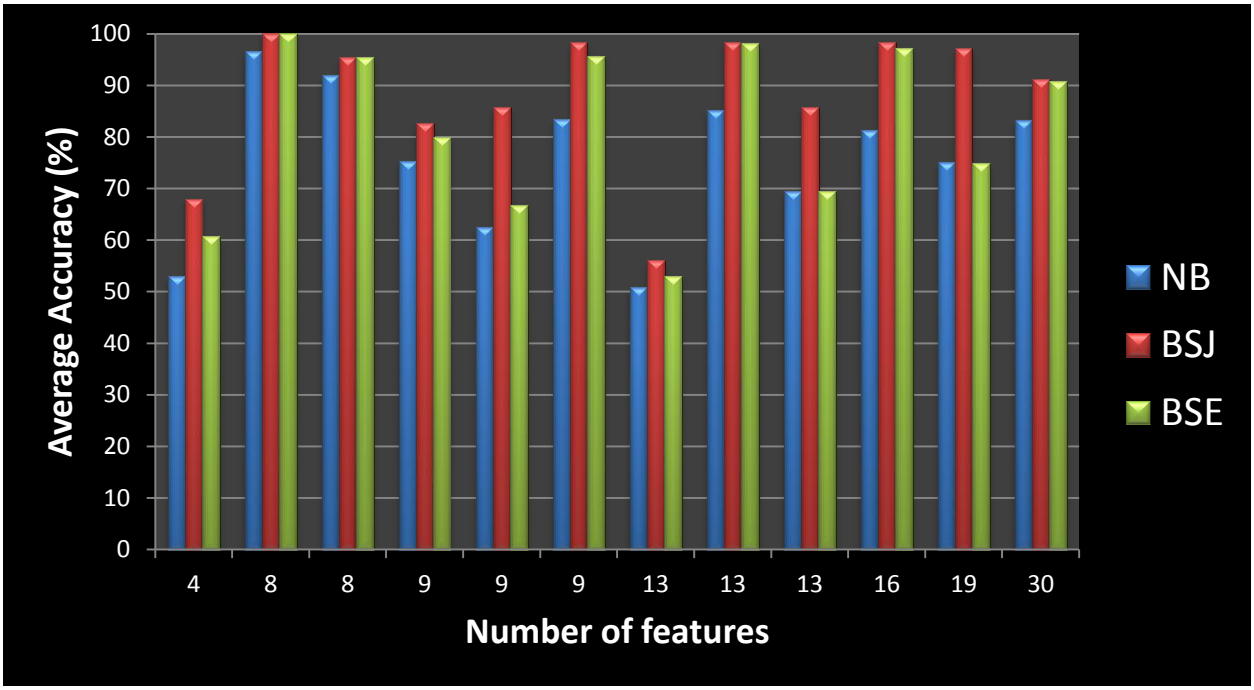

Figure 9 : Bar Graph of Features vs. Average Accuracy

Figure 9 represents a bar graph where the x-axis represents the number of features in increasing order and the y-axis represents the average accuracy corresponding to the dataset. There is no significant trend as the number of features of the datasets increase.

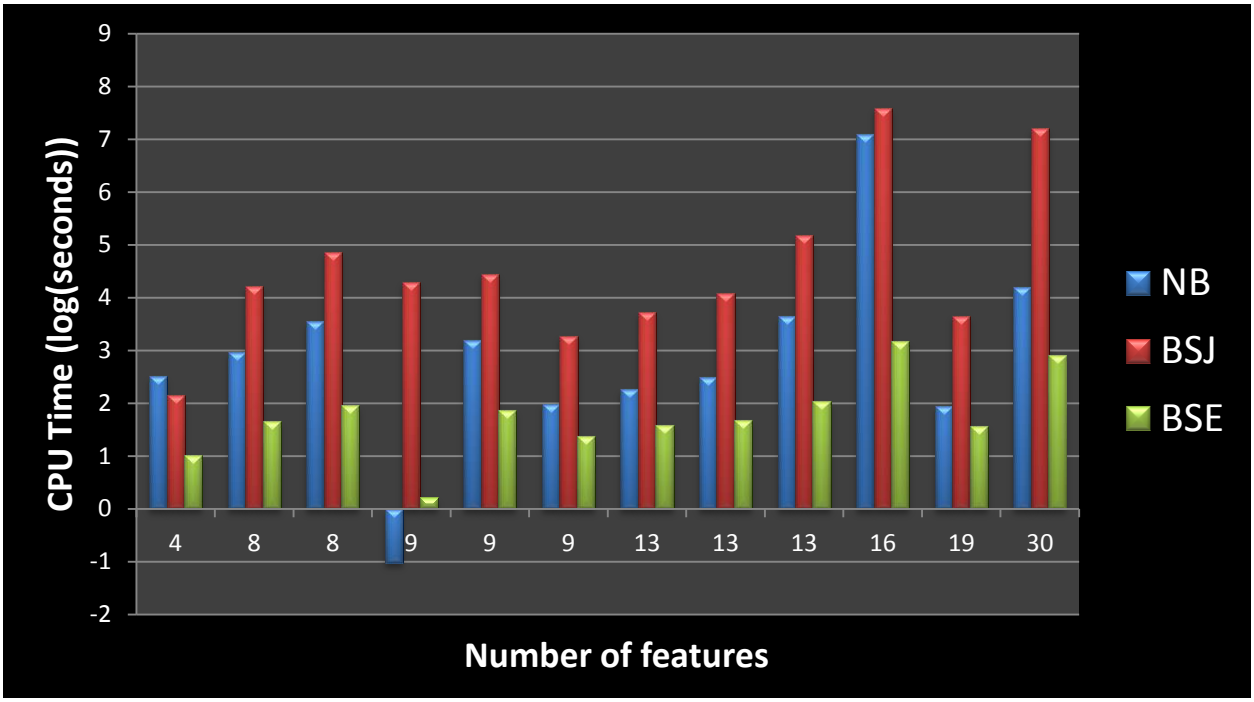

Figure 10 : Bar Graph of Features vs. CPU Time

Figure 10 represents a bar graph where the x-axis represents the number of features and the y-axis represents the log of CPU time. Here we observe that as the number of features increase the CPU time also increases. The dataset having 16 features shows a significant increase in CPU time as compared to the dataset having 19 features. Once reason for this could be the number of classes in each of the datasets. Thus, the CPU time also depends on the number of classes.

6.6 Conclusion

The BSJ algorithm is significantly more accurate than the NB and the BSE algorithm as indicated by the paired t-test results. While BSE performs better than the NB, it does not outperform BSJ in any of the result sets. This leads us to believe that, attributes used in some common databases are not conditionally independent. The violation of the

conditional independence assumption influences the accuracy of the classifier mostly resulting in a linear increase.

Though BSJ notably results in an increase in the accuracies, it is not very efficient when the number of features increase. The computation time for evaluating the BSJ algorithm increases for databases having large number of attributes. Joining mostly happens between more than two attributes, but multiple steps are required. Another drawback with BSJ is that only discretized data can be joined. Hence, continuous valued attributes need to be converted into discrete data which results in loss of information.

We have evaluated the three algorithms namely, NB, BSJ and BSE, on the same datasets, and concluded that BSJ and BSE perform better in databases where NB performed poorly. This suggests that the attributes in those datasets are conditionally dependent among themselves.

REFERENCES

[1] D. Koller and N. Friedman. “Motivation,” in Probabilistic Graphical Models: Principles and Techniques, Massachusetts: MIT Press, pp 1, 2009.

[2] H. Chan. (2005). Sensitivity Analysis of Probabilistic Graphical Models [Online], Available: http://blog.adnanmasood.com/2012/04/27/tagcloud-for-sensitivity-analysis-of-probabilistic-graphical-models/

[3]Probability Distributions. Retrieved April 21, 2014, http://www.wyzant.com/resources/lessons/math/statistics_and_probability/probability_distributions/distributions

[4] R. Zacharski, “Naïve Bayes and Probability Density Functions”, in A Programmer’s Guide to Data Mining, 2012, ch. 6, pp 6–24.

[5] Zwillinger, D., Kokoska, S. (2000) CRC Standard Probability and Statistics Tables and Formulae, CRC Press. ISBN 1-58488-059-7 page 31.

[6] Random Variables [Online], Retrieved April 21, 2014, http://www.stat.yale.edu/Courses/1997-98/101/ranvar.htm

[7] D. Koller. (2013, April). Semantics and Factorization [Online], Available: http://spark-university.s3.amazonaws.com/stanford-pgm/slides/2.1.1-Repn-BNs-semantics.pdf

[8] D. Koller and N. Friedman. “Bayesian Networks,” in Probabilistic Graphical Models: Principles and Techniques, Massachusetts: MIT Press, pp 1, 2009.

[9] D. Koller. (2013, April). Reasoning Patterns [Online], Available: http://spark-university.s3.amazonaws.com/stanford-pgm/slides/2.1.2-Repn-BNs-patterns.pdf

[10] K. M. Leung. (2007, November 28). Naïve Bayes Classifier [Online], Available: http://cis.poly.edu/~mleung/FRE7851/f07/naiveBayesianClassifier.pdf

[11] S. Sayad. (2013). Naïve Bayesian [Online]. Available: http://www.saedsayad.com/naive_bayesian.htm

[12] D. Barber (2001), Learning from Data 1 Naïve Bayes [Online], Available: http://users.cecs.anu.edu.au/~daa/courses/GSAC6017/naivebayes.pdf

[13] M. Pazzani, “Searching for Dependencies in Bayesian Classifiers”, in Proceedings of the Fifth Int. Workshop on AI and Statistics, 1995, pp 2–5.

[14] I. Guyon and A. Elisseeff, “An Introduction to Variable and Feature Selection” in Journal of Machine Learning Research 3, 2003, pp 1157–1182.

[15] Kohavi and G.John, “Wrappers for feature selection”, in Artificial Intelligence, Boston, 1996, pp 2.

[16] M. Pazzani, “Constructive Induction of Cartesian Product Attributes” in ISIS: Information, Statistics and Induction in Science. Singapore, pp 66–77.

[17] S. Raaijmakers, “Finding Representations for Memory-Based Language Learning”, The Netherlands, 1999.

[18] Bayesian Classification tool. Retrieved March 25, 2014, from http://sourceforge.net/projects/weka/

[19] Datasets for cross validation. Retrieved March 25, 2014, from http://www.keel.es/

[20] Datasets for Weka Tool. Retrieved April 2, 2014, from http://www.cs.waikato.ac.nz/ml/weka/datasets.html

[21] UCI Datasets for cross validation. Retrieved March 25, 2014, from https://archive.ics.uci.edu/ml/datasets.html

[22] The Statistics Centre. Mathematical and Statistical Science of University of Alberta [Online]. Retrieved March 27, 2014 from http://www.stat.ualberta.ca/~hooper/teaching/misc/Pvalue.pdf