**8 .Demonstration of classification rule process on dataset employee.ar ff using naïve**

**bayes algorithm**

**Aim:** This experiment illustrates the use of naïve bayes classifier in weka. The sample data

set used in this experiment is “employee”data available at arff format. This document

assumes that appropriate data pre processing has been performed.

**5.3 THEORY:**

Naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even though these features depend on the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple. An advantage of the naive Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix The

**Naive Bayes Probabilistic Model :**

The probability model for a classifier is a conditional model

P(C|F1 .................Fn) over a dependent class variable *C* with a small number of outcomes or *classes*, conditional on several feature variables *F*1 through *Fn*. The problem is that if the number of features *n* is large or when a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable.

Using Bayes‟ theorem, we write P(C|F1...............Fn)=[{p(C)p(F1..................Fn|C)}/p(F1,........Fn)]

In plain English the above equation can be written as Posterior= [(prior \*likehood)/evidence]

In practice we are only interested in the numerator of that fraction, since the denominator does not depend on *C* and the values of the features *Fi* are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model p(C,F1........Fn) which c an be rewritten as follows, using repeated applications of the definition of conditional probability:

p(C,F1........Fn) =p(C) p(F1............Fn|C) =p(C)p(F1|C) p(F2.........Fn|C,F1,F2)

=p(C)p(F1|C) p(F2|C,F1)p(F3.........Fn|C,F1,F2)

= p(C)p(F1|C) p(F2|C,F1)p(F3.........Fn|C,F1,F2)......p(Fn|C,F1,F2,F3.........Fn1)

Now the "naive" conditional independence assumptions come into play: assume that each feature *Fi* is conditionally independent of every other feature *Fj* for j≠i .

This means that p(Fi|C,Fj)=p(Fi|C) and so the joint model can be expressed as p(C,F1,.......Fn)=p(C)p(F1|C)p(F2|C)...........= p(C)π p(Fi|C) This means that under the above independence assumptions, the conditional distribution over the class variable *C* can be expressed like this:

p(C|F1..........Fn)= p(C) πp(Fi|C) Z

where *Z* is a scaling factor dependent only on F1.........Fn, i.e., a constant if the values of the feature variables are known. Models of this form are much more manageable, since they factor into a so called *class prior p*(*C*) and independent probability distributions p(Fi|C). If there are *k* classes and if a model for each p(Fi|C=c) can be expressed in terms of *r* parameters, then the corresponding naive Bayes model has (*k* - 1) + *n r k*

parameters. In practice, often *k* = 2 (binary classification) and *r* = 1 (Bernoulli variables as features) are common, and so the total number of parameters of the naive Bayes model is 2*n* + 1, where *n* is the number of binary features used for prediction

P(h/D)= P(D/h) P(h) P(D)

• P(h) : Prior probability of hypothesis h

• P(D) : Prior probability of training data D

• P(h/D) : Probability of h given D

• P(D/h) : Probability of D given h

Naïve Bayes Classifier : Derivation

• D : Set of tuples

– Each Tuple is an „n‟ dimensional attribute vector

– X : (x1,x2,x3,…. xn)

• Let there me „m‟ Classes : C1,C2,C3…Cm

• NB classifier predicts X belongs to Class Ci iff

– P (Ci/X) > P(Cj/X) for 1<= j <= m , j <> i

• Maximum Posteriori Hypothesis

– P(Ci/X) = P(X/Ci) P(Ci) / P(X)

– Maximize P(X/Ci) P(Ci) as P(X) is

constant Naïve Bayes Classifier : Derivation

• With many attributes, it is computationally expensive to evaluate P(X/Ci)

• Naïve Assumption of “class conditional independence”

• P(X/Ci) = n P( xk/ Ci)10

k = 1

• P(X/Ci) = P(x1/Ci) \* P(x2/Ci) \*…\* P(xn/ Ci)

**Steps involved in this experiment:**

1. We begin the experiment by loading the data (employee.arff) into weka.

**Step2**: next we select the “classify” tab and click “choose” button to select the “id3”classifier.

**Step3:** now we specify the various parameters. These can be specified by clicking in the text

box to the right of the chose button. In this example, we accept the default values his default version does perform some pruning but does not perform error pruning.

**Step4:** under the “text “options in the main panel. We select the 10-fold cross validation as

our evaluation approach. Since we don’t have separate evaluation data set, this is necessary toget a reasonable idea of accuracy of generated model.

**Step-5:** we now click”start”to generate the model .the ASCII version of the tree as well as

evaluation statistic will appear in the right panel when the model construction is complete.

**Step-6:** note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the

classification)

Step-7: now weka also lets us a view a graphical version of the classification tree. This can be

done by right clicking the last result set and selecting “visualize tree” from the pop-up menu.

**Step-8:** we will use our model to classify the new instances.

**Step-9:** In the main panel under “text “options click the “supplied test set” radio button and

then click the “set” button. This will show pop-up window which will allow you to open the

file containing test instances.