**Feedback System using Bayesian Classifier**

*This report is submitted for the evaluation of assignment component for the subject "Data Mining" with course code CSPE01 during the term August to December 2015.*

Submitted by

**DEEPAK JAYAPRAKASH-1MS13CS037**

**MAHEK SALUJA – 1MS13CS061**

**NIHAL BARICK - 1MS13CS071**

Faculty Incharge :

**D. Pradeep Kumar**

Assistant Professor

Dept. of CSE M.S.Ramaiah Institute of Technology

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

M.S.RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)

BANGALORE-560054

www.msrit.edu

August to December 2015

**ABSTRACT**

In this project we are implementing the feedback system using Naïve Bayes theorem,using the principle of predicting the output given the test cases given in the Data Set.We either predict whether the feedback is good or bad.

The training set is limited but the accuracy of the algorithm is pretty efficient with just so many tuples. Then the input to the machine is the feedback forms which has 4 attributes i.e. 4 questions and each can take 3 values.

Then we classify the input into a good feedback or bad feedback by calculating the conditional probabilities.

**Algorithm**

Naive Bayesian Classiﬁcation The Naïve Bayesian classiﬁer, or simple Bayesian classiﬁer, works as follows:

1. Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X=(x1, x2,..., xn), depicting n measurements made on the tuple from n attributes, respectively, A1, A2,..., An.
2. Suppose that there are m classes, C1, C2,..., Cm. Given a tuple, X, the classiﬁer will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the na¨ıve Bayesian classiﬁer predicts that tuple X belongs to the class Ci if and only if **P(Ci|X) > P(Cj|X) for 1≤j≤m,j6=i**. Thus, we maximize P(Ci|X). The class Ci for which P(Ci|X) is maximized is called the maximum posteriori hypothesis. By Bayes’ theorem (Eq. 8.10),

**P(Ci|X)=P(X|Ci)P(Ci) P(X)**

3. As P(X) is constant for all classes, only **P(X|Ci)P(Ci)** needs to be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, P(C1)=P(C2)=···=P(Cm), and we would therefore maximize P(X|Ci). Otherwise, we maximize P(X|Ci)P(Ci). Note that the class prior probabilities may be estimated by **P(Ci)=|Ci,D|/|D|**,

where|Ci,D|is the number of training tuples of class Ci in D.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute P(X|Ci). To reduce computation in evaluating P(X|Ci), the na¨ıve assumption of class-conditional independence is made. This presumes that the attributes’ values are conditionally independent of one another, given the class label of the tuple (i.e., that there are no dependence relationships among the attributes). Thus,

**P(X|Ci)=∏P(xk|Ci)** **=P(x1|Ci)×P(x2|Ci)×···×P(xn|Ci).**

Dataset:

static char workload[]= {'v','v','a','n','v','a','n','v','a','n','n','n','v','a','v','v','v','n','v','a','v','a','v','a','n','v'};

static char technical[]= {'v','v','a','a','a','a','n','a','v','n','n','a','a','a','v','v','v','v','n','a','a','n','n','v','a','a'};

static char neccessity[]={'v','n','v','a','n','a','a','v','n','n','n','a','v','v','a','a','v','v','v','n','n','v','n','v','v','a'};

static char advance[]= {'v','a','n','n','n','v','a','a','a','a','a','n','a','a','n','v','n','v','v','v','n','v','n','n','n','n'};

static char class1[]= {'A','A','B','B','B','A','B','A','B','B','B','B','A','A','A','A','A','A','A','B','B','A','B','A','B','A'};

static double prob[][]=new double[4][2];

OUTPUT

