## Naïve Bayes Algorithm Used –

A naive Bayes classiﬁer (NBC) is a simple probabilistic based method that may expect the magnificence club possibilities. A naive Bayes classiﬁer can easily manage missing characteristic values by absolutely omitting the corresponding probabilities for those attributes while calculating the probability of club for every class. It also calls for the magnificence conditional independence, i.e., a characteristic on a given magnificence is independent of these of different attributes.

**Data :** ‘Training dataset: D = X1,X2,…Xn // Training dataset, D, which contains a set of training instances and their associated class labels.

**Result :** noise list: *noiselist*

**for each** class, *Ci ε D* **do**

Find the prior probabilities, *P(Ci)*

# end

**for each** attribute values, *Ai ε D* **do**

Find the class of conditional probabilities, *P(Ai | Ci)*

# end

**for each** class, *Xi ε D* **do**

Find the conditional probabilities, *P(Xi | Ci) if P(Xi | Ci) == 0* **then**

**//** Use Laplacian Estimator recalculate the conditional probability *P(Xi | Ci)* using Laplacian Estimator

# end

**if *Xi*** is misclassified **then**

*misClasslist <-- Xi*

*misProlist <-- P(Xi | Ci) // Store all the probabilities for all misclassified*

# end else

**end**

*pureClasslist <-- Xi*

*pureProlist <-- P(Xi | Ci) // Store the probabilities for all purely classified instances.*

# end

*Tnoise = find****MIN*** *(pureProlist) // Use as noise threshold*

**for each** instance xi *ε misClasslist* **do**

Find the conditional probablities *P(Xi | Ci) misProlist*

if *P(Xi | Ci) < Tnoise* **then**

*noiselist <--* Xi // *Store the instance as noise*

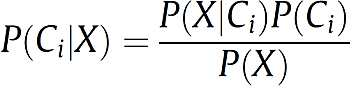
# end

**end**

return *noiselist*

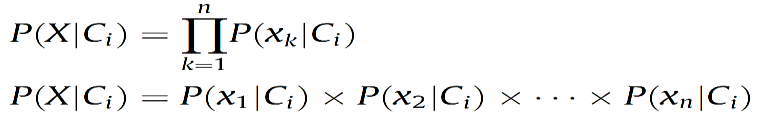
Algorithm 1 : Noise Detection

Let D be a training set of data instances and their associated class labels. Each instance is represented by an n-dimensional attribute vector, X ={x1,x2,...,xn}, depicting n measurements made on the instance from n attributes, respectively,{A1,A2,...,An}.

Suppose that there are m classes,{C1,C2,...,Cm}. For a test instance, X, the classiﬁer will predict that X belongs to the class with the highest conditional probability, conditioned on X. That is, the naive Bayes classiﬁer predicts that the instance X belongs to the class Ci, if and only if P(Ci|X) > P(Cj|X) for 1 ≤ j ≤ m,j 6= i The class Ci for which P(Ci|X) is maximized is called the Maximum Posteriori Hypothesis.

In Bayes theorem shown in Equation (1), as P(X) is a 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 likely equal, that is, P(C1) = P(C2) = ... = P(Cm), and therefore we would maximize P(X|Ci).

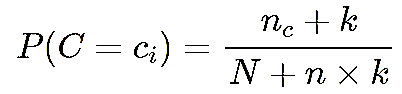
Otherwise, we maximize P(X|Ci)P(Ci). The class prior probabilities are calculated by P(Ci) = |Ci,D|/|D|, where |Ci,D| is the number of training instances of class Ci in D. To compute P(X|Ci) in a dataset with many attributes is extremely computationally expensive. Thus, the naive assumption of class-conditional independence is made in order to reduce computation in evaluating P(X|Ci). This presumes that the attributes’ values are conditionally independent of one another, given the class label of the instance, i.e., there are no dependence relationships among attributes. Thus, Equation (2) and (3) are used to produce P(X|Ci).

…Eq.2

P(X|Ci) = P(x1|Ci)×P(x2|Ci)×...×P(xn|Ci) (3) In Equation (2), xk refers to the value of attribute Ak for instance X. Therefore, these probabilities P(x1|Ci),P(x2|Ci),...,P(xn|Ci) can be easily estimated from the training instances.

If the attribute value, Ak, is categorical, then P(xk|Ci) is the number of instances in the class Ci ∈ D with the value xk for Ak, divided by |Ci,D|, i.e., the number of instances belonging to the class Ci ∈ D. To predict the class label of instance X,P(X|Ci)P(Ci) is evaluated for each class Ci ∈ D. The naive Bayes classiﬁer predicts that the class label of instance is the class Ci, if and only if P(X|Ci)P(Ci) > P(X|Cj)P(Cj) for 1 ≤ j ≤ m and j 6= i In other words, the predicted class label is the class Ci for which P(X|Ci)P(Ci) is the maximum.

## Laplacian Estimation -



As in naive Bayes classifier discussed above, we calculate **P(X|Ci)** as the product of the probabilities **P(x1|Ci)×P(x2|Ci)×...×P(xn|Ci),** based on the independence assumption and class conditional probabilities, and end up with a actual probable value of zero for some **P(x|Ci).** This may happen if the attribute value x is never observed in the training data for a particular class **Ci.** Therefore, the generated Equation [3](https://www.groundai.com/project/a-machine-learning-based-robust-prediction-model-for-real-life-mobile-phone-data/1#S4.E3) becomes zeros for such attribute value regardless the values of other attributes. Thus, naive Bayes classifier is unable to predict the class of such test instances. In order to resolve such issues in our model, we use Laplace estimator *Cestnik (1990)* to scale up the values by smoothing factor. In Laplace-estimate, the class probability is defined as *Cestnik (1990)*:

…Eq.3

Where nc is the number of instances satisfying C=ci, N is the number of training instances, n is the number of classes and k=1. Let’s consider a phone call behavior example, for the behavior class ‘reject’ in the training data containing 1000 instances, we have 0 instance with relationship is qual to unknown, 990 instances with relationship=friend, and 10 instances with relationship is equal to mother.

The probabilities of these contexts are 0, 0.990 (from 990/1000), and 0.010 (from 10/1000), respectively. On the alternative hand in keeping with equation 2 the probabilities of those contexts could be as follows:

In this way we obtain the above non zero possibilities rounded up to a three decimal places severally victimization Laplace calculator outlined on top of these.

The “new” probability estimates are close to their “previous” counterparts, and these values can be used for further processing in our model.

## Noise Detection Technique -

In this section, to discuss the noise detection technique in order to detect phone call, Machine running, behavior. Figure 1 shows the complete execution of noise detection system block and technique used. In order to detect noise, The implemented system use Naive Bayes classiﬁer (NBC)[9] as the basis for noise identiﬁcation. Using nbc we tend to rst calculate the probability for every attribute by scanning the coaching knowledge, The portable dataset Each example contains 4 attribute values, time, area and relationship and corresponding noise level behavior. For each attribute value, respectively for this dataset. Using these possibilities we calculate the conditional possibility for each example as nbc was implemented beneath the independence.

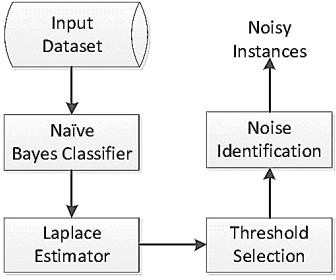


Fig. 8 : Input Phase

Assumption, it estimates zero possibilities if the probability for one attribute is zero attribute is zero. In such cases, we use Laplace- estimator [10] estimate the conditional probability of any of the attribute value. After the match value prediction it shows the “Number of record scans” and show the probability generated. Once we have calculated conditional probability for each instance, we diﬀerentiate between the purely classiﬁed instances and misclassiﬁed instances using machine learning.