BANK MARKETING

DATA MINING GROUP PROJECT

Group7

Arun Kumar Narayana Murthy

Manikandan Sundarapandian

Muthu Kannan Subramaniam

Sathya Narayanan Manivannan

Sourabh Mahajan

Contents

INTRODUCTION

The Data Mining Process is a powerful technique that helps not only in decision making based on the data that is available but also helps in predicting the potential change or result that might occur in the future. The Data Mining Technique can come up with various useful predictions that usually cannot be interpreted using the graphical reporting.

Classification is one of the data mining techniques that help in classifying the items according to the items with predefined set of classes. In this project, we are implementing the Classification technique in predicting the 'Subscription' attribute based on the other relevant fields.

This paper will include evaluation of three different algorithms using the tool WEKA. The Data that is collected will enable better strategies for finding possible customers who will subscribe for term deposit.

Domain description

Bank Marketing campaigns are dependent on customers' huge electronic data. Identifying customers who are more likely to respond to new offers is an important issue in Direct Marketing. With the huge amount of data, it is impossible for analysts to come up with interesting information about the customers. In direct marketing, data mining has been used extensively to identify potential customers for a new offer (target selection).
The data is related with direct marketing campaigns (phone calls) of a Portuguese banking institution. The classification goal is to predict if the client will subscribe a term deposit (values y or n).

Problem statement

The requirement is to predict the value of Subscription attribute based on Job Type, Client Marital Status, Education, Bank Balance, Housing Loan, Personal Loan, Contact, Last Contact Day, Last Contact, Month, Last Contact Duration, Current Campaigns, Days Passed, Previous Campaigns and Previous Outcome.

DATA SET

Description

The Dataset includes a total of 61,079 rows and 16 attributes containing both Nominal and Numeric attribute types like Job Type, Client Marital Status, Education, Bank Balance, Housing Loan, Personal Loan, Contact, Last Contact Day, Last Contact, Month, Last Contact Duration, Current Campaigns, Days Passed, Previous Campaigns, Previous Outcome and

Subscription Status of several customers, whose information was extracted from a Portuguese Banking Institution.

Dataset was obtained from UCI Website.
http://archive.ics.uci.edu/ml/datasets/Bank+Marketing

Attribute Description

Attributes	Description	Values
Age	Age of the Client	Numeric Value
Job	Type of job of the client	blue-collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed, unknown.
Marital Status	Marital Status of the client	divorced, married, single, unknown
Education	Educational qualification of the client	Basic 4y, basic 6y, basic 9y, high school, illiterate, professional course, university degree, unknown
Bank Balance	Bank balance of the client	no, yes, unknown
Housing Loan	Whether the client has housing loan	no, yes, unknown
Personal Loan	Whether the client has personal loan	no, yes
Contact	Contact Communication type	Cellular, Telephone
Last Contact Day	Client's Last Contact day of the week	Mon, Tue, Wed, Thu, Fri
Last Contact Month	Client's Last Contact month of the year	Jan, Feb, Mar…Nov, Dec
Last contact duration	Client's Last contact duration, in seconds(numeric)	Call Duration in Seconds
Current Campaigns	Number of contacts performed during this campaign and for this client	Numeric Data
Days Passed	Number of days that passed by after the client was last contacted from a previous campaign	Numeric data (999 means client was not previously contacted)
Previous Campaigns	number of contacts performed before this campaign and for this client (numeric)	Numeric Data
Previous Outcome	outcome of the previous marketing campaign	failure, nonexistent, success, unknown
Subscription	whether the client has subscribed a term deposit	Yes, No

PRE-PROCESSING STEPS

Quality data provides quality decisions. Data preprocessing transforms the data into a format that will be more easily and effectively processed by the algorithm. Real world data are incomplete, noisy, and inconsistent. There are attributes which are false predictors and has missing values, noise, error, other data discrepancies.

Data Cleaning

Data cleaning is a process used to determine inaccurate, incomplete, or unreasonable data and then improve the quality through correcting detected errors and omissions. Raw data is highly susceptible to noise, missing values and inconsistency. The quality of data affects the data mining results. To help improve the quality of data and consequently of mining results, raw data is pre-processed so as to improve the efficiency and ease of mining process. Data pre-processing is one of the most critical steps in a data mining process which deals with preparation and transformation of the initial data set.

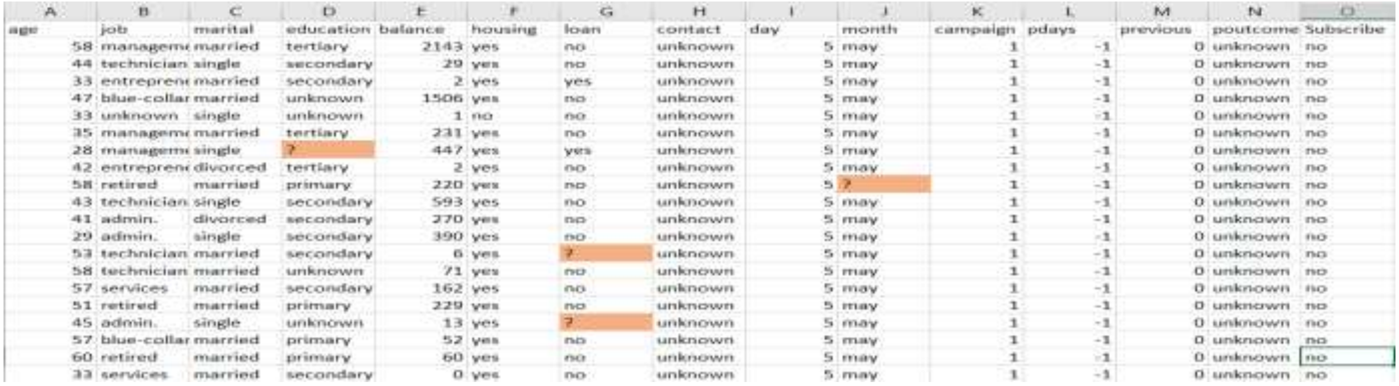
Missing Values

Real-world data tends t'o be incomplete, noisy and inconsistent. An important task when preprocessing the data is to fill in the missing values, smooth out noise and correct inconsistencies.

Some of the steps to handle Missing Values are as follows:

- Ignore the data row
- Use a global constant to fill in for missing values
- Use attribute mean for all samples belonging to the same class
- Use a data mining algorithm to predict the most probable value

In our dataset, of the 61079 instances, there were a total of 105 Instances with a missing value in either of their attributes. These missing data have been represented with a "?". These missing values were replaced using unsupervised field in 'Filters' option.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
age	job	marital	education	balance	housing	loan	contact	day	month	campaign	pdays	previous	poutcome	Subscribe
58	managem	married	tertiary	2143	yes	no	unknown	5	may	1	-1	0	unknown	no
44	technician	single	secondary	29	yes	no	unknown	5	may	1	-1	0	unknown	no
33	entrepren	married	secondary	2	yes	yes	unknown	5	may	1	-1	0	unknown	no
47	blue-collar	married	unknown	1506	yes	no	unknown	5	may	1	-1	0	unknown	no
33	unknown	single	unknown	1	no	no	unknown	5	may	1	-1	0	unknown	no
35	managem	married	tertiary	231	yes	no	unknown	5	may	1	-1	0	unknown	no
28	managem	single	?	447	yes	yes	unknown	5	may	1	-1	0	unknown	no
42	entrepren	divorced	tertiary	2	yes	no	unknown	5	may	1	-1	0	unknown	no
58	retired	married	primary	220	yes	no	unknown	5	?	1	-1	0	unknown	no
43	technician	single	secondary	593	yes	no	unknown	5	may	1	-1	0	unknown	no
41	admin.	divorced	secondary	270	yes	no	unknown	5	may	1	-1	0	unknown	no
29	admin.	single	secondary	390	yes	no	unknown	5	may	1	-1	0	unknown	no
53	technician	married	secondary	6	yes	?	unknown	5	may	1	-1	0	unknown	no
58	technician	married	unknown	71	yes	no	unknown	5	may	1	-1	0	unknown	no
57	services	married	secondary	162	yes	no	unknown	5	may	1	-1	0	unknown	no
51	retired	married	primary	229	yes	no	unknown	5	may	1	-1	0	unknown	no
45	admin.	single	unknown	13	yes	?	unknown	5	may	1	-1	0	unknown	no
57	blue-collar	married	primary	52	yes	no	unknown	5	may	1	-1	0	unknown	no
60	retired	married	primary	60	yes	no	unknown	5	may	1	-1	0	unknown	no
33	services	married	secondary	0	yes	no	unknown	5	may	1	-1	0	unknown	no

Duplicate Values

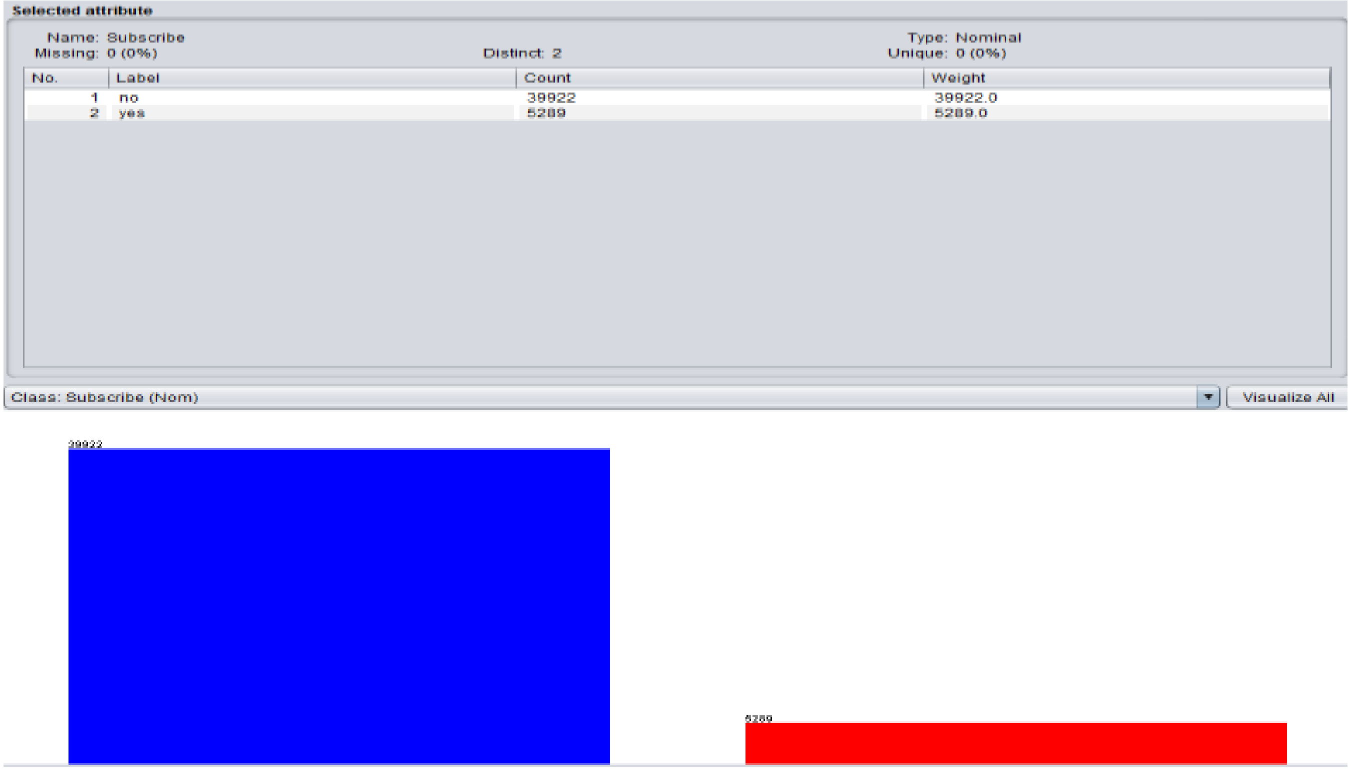
The problem of detecting and eliminating duplicated data is one of the major problems in the broad area of data cleaning. Duplicate elimination is hard because it is caused by several types of errors like typographical errors and different representations of the same logical value. Hence, another important aspect of data cleaning was to check for duplicate values in the dataset. No duplicate values were detected from our Bank Marketing dataset.

Class Imbalance

Learning from imbalanced data has been studied actively for about two decades in machine learning. A vast number of techniques have been tried, with varying results and few clear answers. Data scientists facing this problem have no definite answer since it entirely depends on the data.
Let us see some of the useful approaches to handle Class Imbalance.

- Do nothing. Sometimes you get lucky and nothing needs to be done. You can train on the so-called *natural* (or *stratified*) distribution and sometimes it works without need for modification.
- Balance the training set in some way(Smote):
    - Oversample the minority class.
    - Under sample the majority class.
    - Synthesize new minority classes.
- Throw away minority examples and switch to an anomaly detection framework.
- Construct an entirely new algorithm to perform well on imbalanced data.

We have used SMOTE technique to balance our class attribute.



Selected attribute

Name: Subscribe
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	no	39922	39922.0
2	yes	5289	5289.0

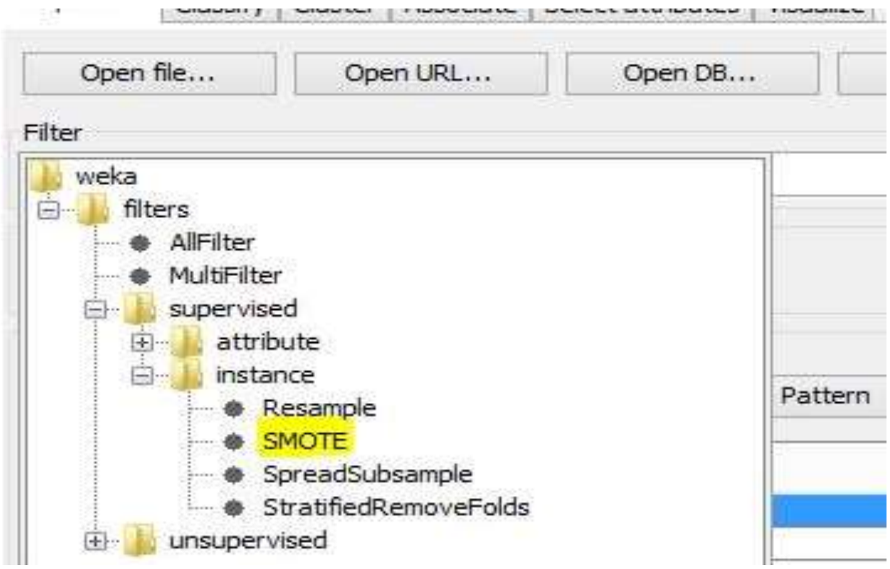
Class: Subscribe (Nom)

According to our dataset, as we can see from the above image the class attribute has a huge imbalance. Applying an algorithm over this dataset might build models which are biased towards one value of the class variable. Hence, we have chosen the SMOTE filter option (i.e., by oversampling the minority Class) to handle the Class Imbalance problem.

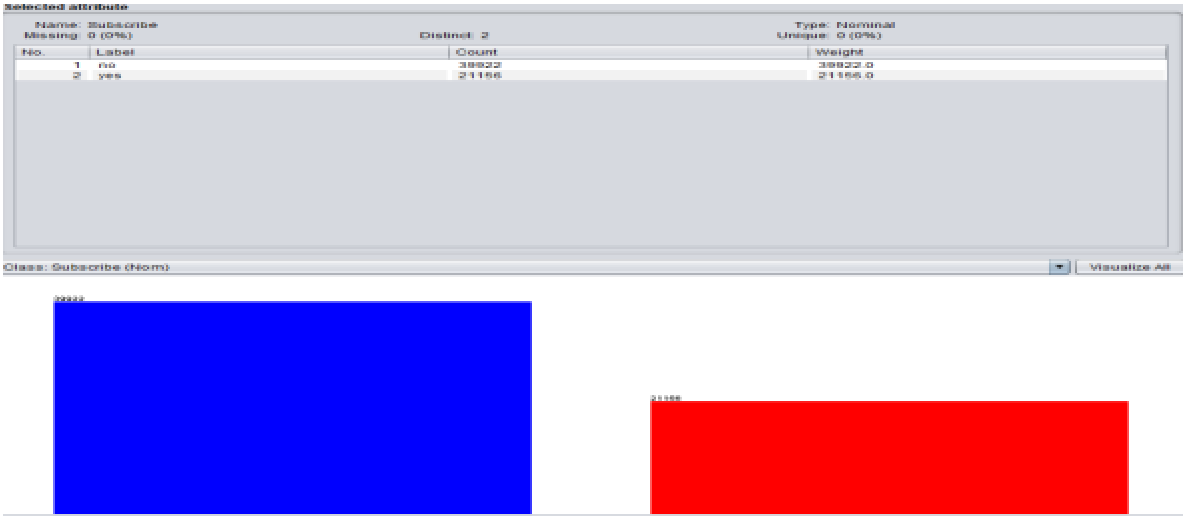
What is SMOTE? What does it do?

- Resamples a dataset by applying the Synthetic Minority Oversampling Technique (SMOTE).
- SMOTE option does oversampling of the minority class, i.e., adds additional instances where the minority class can be oversampled. Similarly, down sampling (or under-sampling) the majority class could also rectify the imbalance.

To use SMOTE filter in Weka,



After applying **Smote filter** to our dataset**,** the imbalance data is modified and the class attribute has pretty balanced data.


Selected attribute

Name: Subscribe
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	no	39922	39922.0
2	yes	21156	21156.0

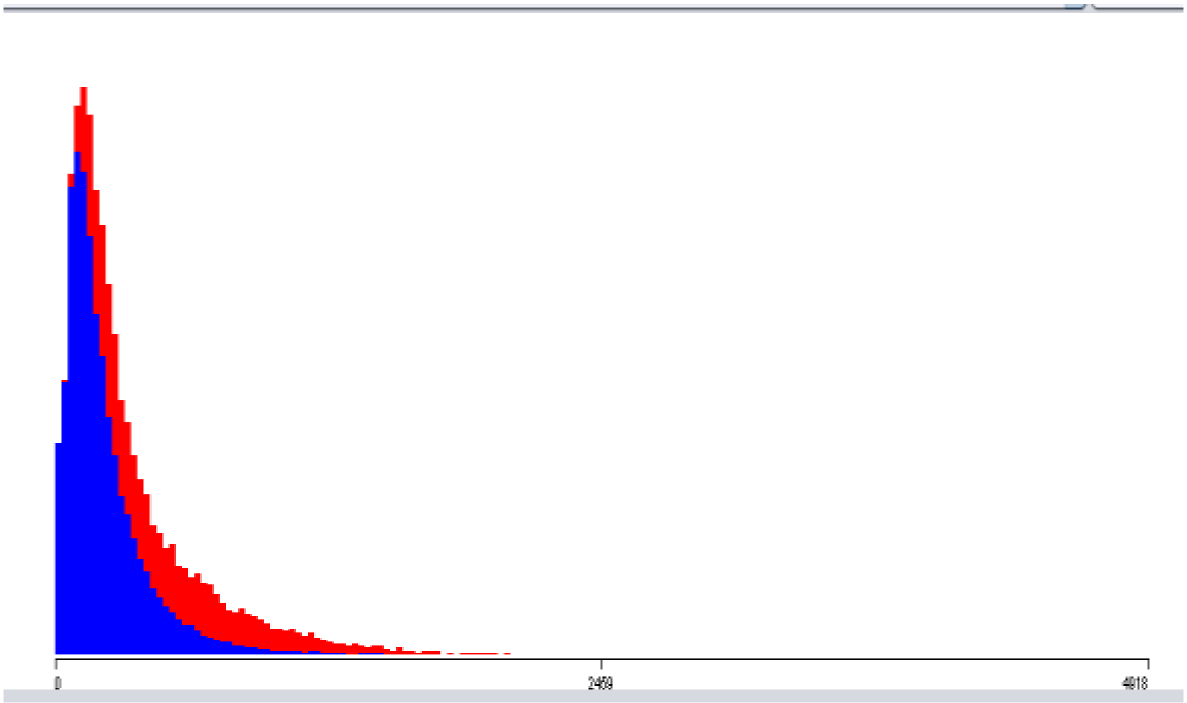
Class: Subscribe (Nom)

Removing Outliers (Skewed Data):

Outliers do not follow the pattern of the majority of the data. Such observations need to be set apart at the onset of any analysis simply because their distance from the bulk of the data ensures that they will exert a disproportionate pull on any model fitted by maximum likelihood.

Furthermore, detecting outliers is a statistical procedure with a well-defined objective and whose efficacy can be measured. It is also important to point out that no matter how they are identified, the outliers of a group of suspect observations can be assessed simply by measuring their influence on a non-robust fit.

While pre-processing our dataset, outliers from the data set which are positively skewed are removed to check for improvement in accuracy of the classifiers. Some of the attributes had skewed data like the one showed below where the values are skewed between (0-1735) even though the range goes up to 4000.

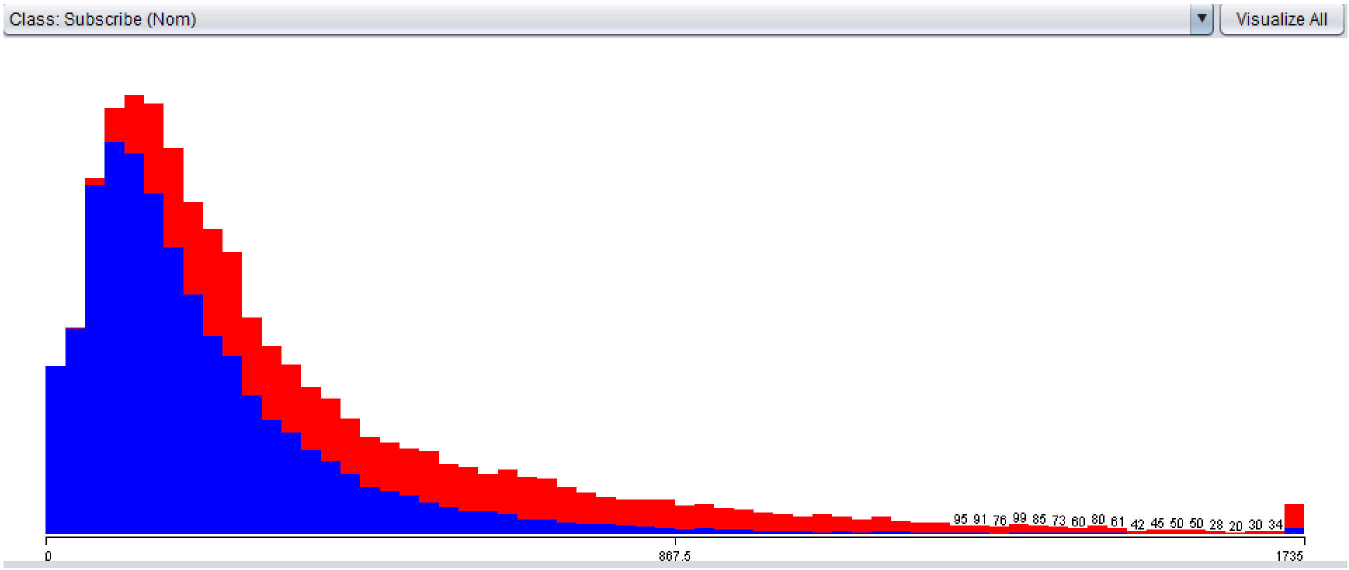


List of Attributes with skewed data:

- Bank Balance
- Last Contact Duration
- Current Campaigns
- Days Passed
- Previous Campaigns

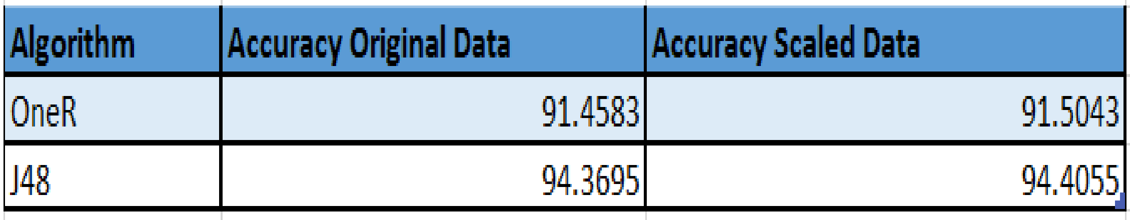
Scaling Data

To handle this problem, we scaled the data by modifying the range for the one of the attributes from (0-1735).



Class: Subscribe (Nom)

After changing the scale, we compared the runs between Original data and scaled data.



Algorithm	Accuracy Original Data	Accuracy Scaled Data
OneR	91.4583	91.5043
J48	94.3695	94.4055

Observation

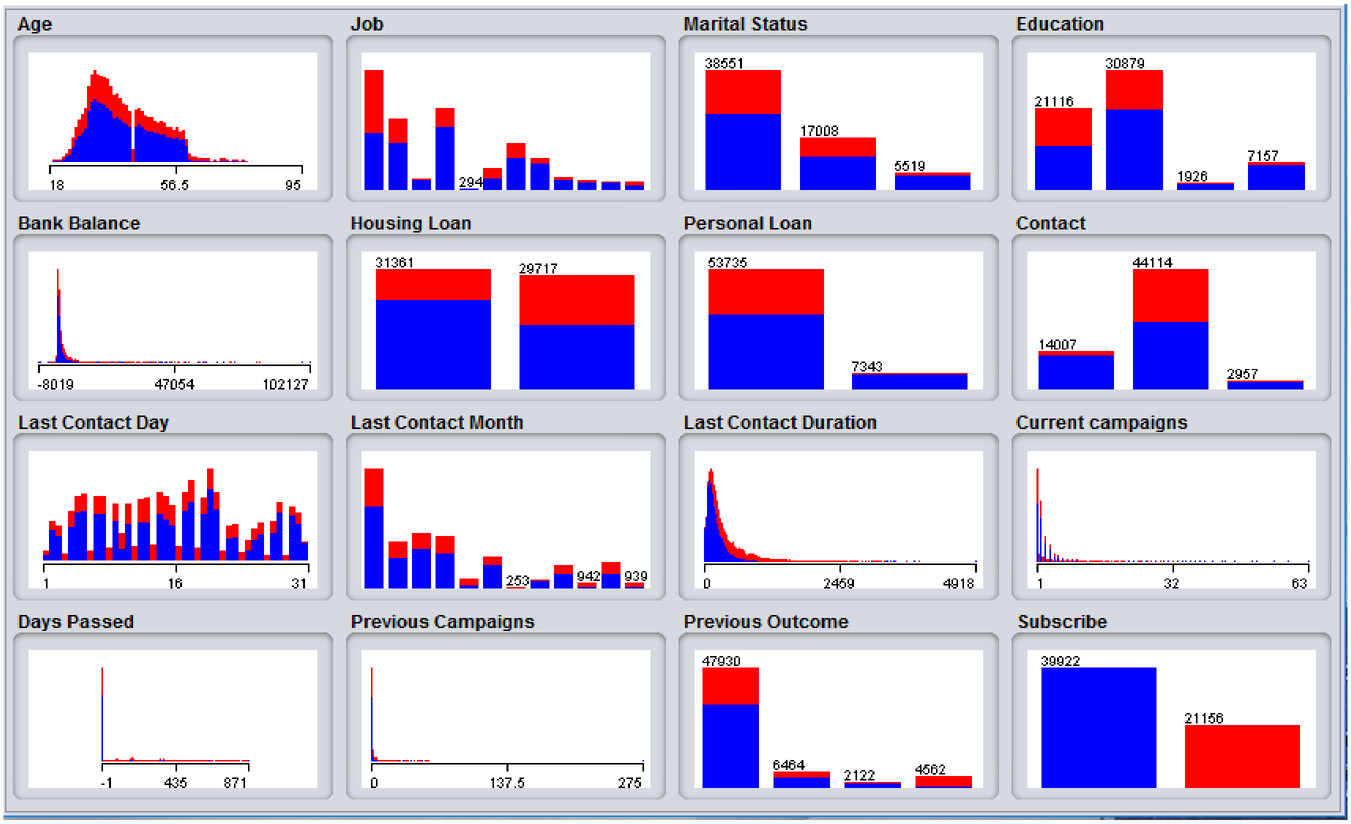
Though the algorithm took less time to come up with the result, there was not much difference in the accuracies between original data and scaled data. Hence, we decided to build algorithms by sticking to the original skewed dataset on which SMOTE was used.

DATA VISUALIZATION

Data visualization is a general term that describes any effort to help people understand the significance of data by placing it in a visual context such as patterns, graphs, trends and correlations that might go undetected in text-based data but can be recognized easily with data visualization software.

Most business intelligence software vendors embed data visualization tools into their products, either developing the visualization technology themselves or sourcing it from companies that specialize in visualization.

The following picture shows a visual representation of the banking dataset that we have chosen:



EXPERIMENT DESIGN

Experiment Design is the best approach for testing our hypothesis. It refers to how participants are allocated to the different combinations in an experiment. The most common way to design an experiment is to divide the participants into two groups, the experimental group, and the control group, and then introduce a change to the experimental group but not the control group.

One member of each matched pair must be randomly assigned to the experimental group and the other to the control group.

In our dataset, following factors are considered for Experimental Design:

- Noise (0%, 10%).
- Size of the Training set (10/90, 80/20).

Now, let us consider the experimental group as

F1- Size of training set
F2- Noise

and the control group as

F11- 10% training set
F12- 80% training set

F21- 0% Noise
F22- 10% Noise

Here, the concept of counterbalancing is applied to these factors and the following four scenarios are created in the experimental design

C1- 0% noise, 10% training set

C2- 0% noise, 80% training set

C3- 10% noise, 10% training set

C4- 10% noise, 80% training set

ALGORITHMS USED

We wanted to use algorithms on probability based, tree based and rule based classifiers. Since noise is effectively managed by Naive Bayes classifier, we have chosen this in probability based classifier category. In the tree based classifier category, we chose J48 algorithm since it gives better accuracy and the results are easily compared with other algorithms. We chose Part algorithm because the algorithm was quite new to all of us and wanted to test the dataset in a non-familiar algorithm so that there is some additional learning to all of us apart from the familiar algorithms. Here are the algorithms/classifiers that we selected.

- Naïve Bayes Classifier
- J48
- Part

Naïve Bayes Classifier:

Bayes rule is applied to calculate the posterior from the prior and the likelihood, because the latter two is generally easier to be calculated from a probability model. We have chosen to work with Naïve Bayes classifier under this method. This method goes by the name of Naïve Bayes because it's based on Bayes' rule and "naïvely" assumes independence—it is only valid to multiply probabilities when the events are independent.

One of the nice things about Naïve Bayes is that missing values are no problem at all. If a value is missing in a training instance, it is simply not included in the frequency counts, and the probability ratios are based on the number of values that occur rather than on the total number of instances. Numeric values are usually handled by assuming that they have a "normal" probability distribution. The advantages of Naïve Bayes are that it only requires a small amount of training data to estimate the parameters necessary for classification. Because independent variables are assumed, only the variables for each class need to be determined and not the entire covariance matrix.

Trees – J48

A decision tree is a predictive machine-learning model that decides the target value (dependent variable) of a new sample based on various attribute values of the available data. The internal nodes of a decision tree denote the different attributes, the branches between the nodes tell us the possible values that these attributes can have in the observed samples, while the terminal nodes tell us the final value (classification) of the dependent variable.
The attribute that is to be predicted is known as the dependent variable, since its value depends upon, or is decided by, the values of all the other attributes. The other attributes, which help in predicting the value of the dependent variable, are known as the independent variables in the dataset.

The J48 Decision tree classifier follows the following simple algorithm. To classify a new item, it first needs to create a decision tree based on the attribute values of the available training data. So, whenever it encounters a set of items (training set) it identifies the attribute that discriminates the various instances most clearly. This feature can tell us most about the data instances so that we can classify them the best is said to have the highest information gain. Now, among the possible values of this feature, if there is any value for which there is no ambiguity, that is, for which the data instances falling within its category have the same value for the target variable, then we terminate that branch and assign to it the target value that we have obtained.

For the other cases, we then look for another attribute that gives us the highest information gain. Hence, we continue in this manner until we either get a clear decision of what combination of attributes gives us a particular target value, or we run out of attributes. If we run out of attributes, or if we cannot get an unambiguous result from the available information, we assign this branch a target value that the majority of the items under this branch possess.

Now that we have the decision tree, we follow the order of attribute selection as we have obtained for the tree. By checking all the respective attributes and their values with those seen in the decision tree model, we can assign or predict the target value of this new instance.
J48 can work with both continuous and discrete data. It does this by specifying ranges or thresholds for continuous data thus turning continuous data into discrete data.
J48 is well known for its capability of building high accuracy models.
The bestselling point of decision trees is their ease of interpretation and explanation. They are also quite fast and popular and the output is human readable.

PART-algorithm

Part algorithm adopts a supervised machine learning algorithm, namely partial decision trees, as a method for feature subset selection. Feature subset selection aims at finding the smallest feature set having the most beneficial impact on machine learning algorithms, i.e. it's prime goal is to identify a subset of features upon which attention should be centered. More precisely, PART exploits the partial decision tree learning algorithm for feature space reduction. It uses separate-and-conquer method. It builds a partial C4.5 (J48) decision tree in each iteration and makes the "best" leaf into a rule. In each iteration, a rule is derived from a pre-pruned decision tree.

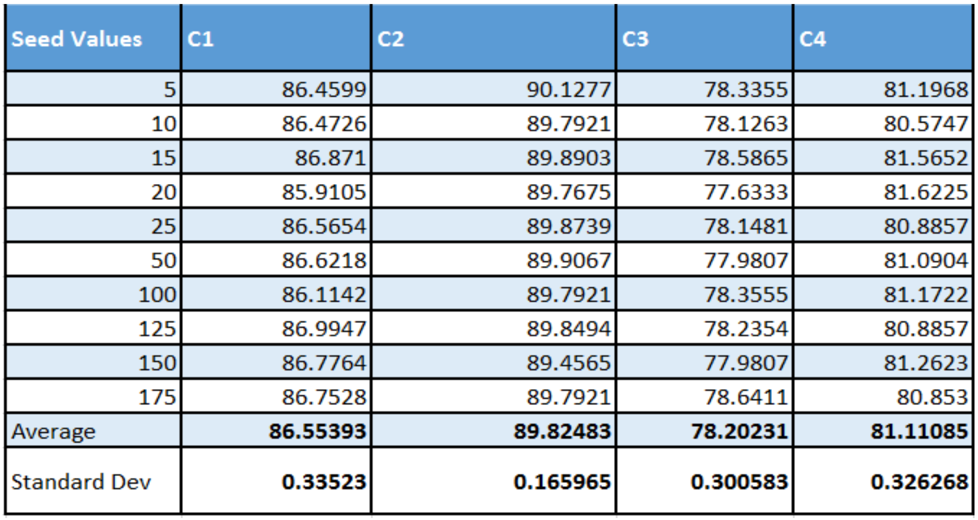
Experimental results

Naïve Bayes Algorithm:

Seed Values	C1	C2	C3	C4
5	81.8737	81.6961	74.6971	74.4925
10	82.1976	81.3278	74.7862	74.6644
15	80.8332	82.6948	74.457	75.6958
20	81.9738	82.2364	74.9154	75.3356
25	81.1042	82.22	73.6784	75.3193
50	82.1957	81.909	75.2174	75.1555
100	82.0993	82.4738	75.3738	75.4175
125	81.3735	82.8422	74.6607	75.8595
150	81.479	82.1955	74.2714	75.6713
175	81.7737	82.572	74.8645	75.704
Average	**81.69037**	**82.21676**	**74.69219**	**75.33154**
Standard Dev	**0.474244**	**0.467721**	**0.481889**	**0.453774**

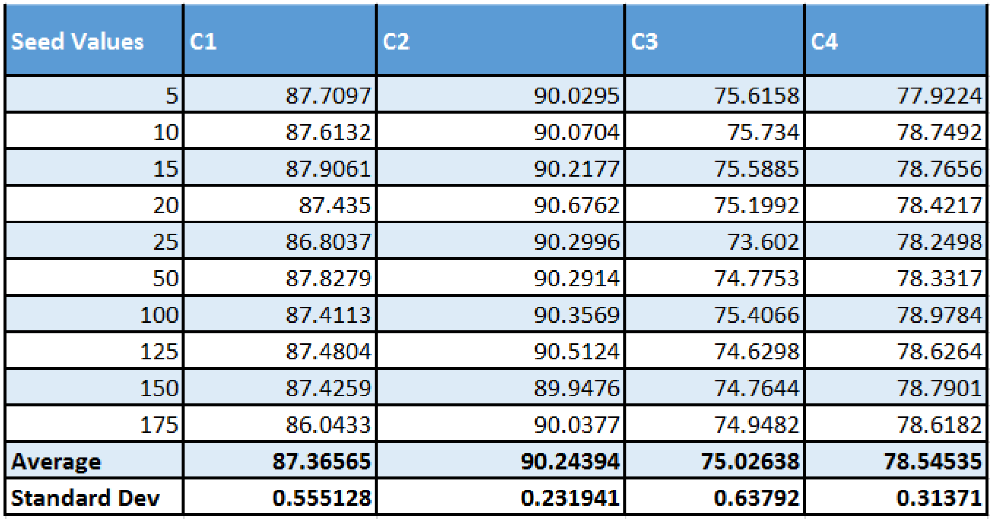
J48 Algorithm:

The top attribute for the J48 algorithm was ‘LAST CONTACT DURATION’ attribute.



Seed Values	C1	C2	C3	C4
5	86.4599	90.1277	78.3355	81.1968
10	86.4726	89.7921	78.1263	80.5747
15	86.871	89.8903	78.5865	81.5652
20	85.9105	89.7675	77.6333	81.6225
25	86.5654	89.8739	78.1481	80.8857
50	86.6218	89.9067	77.9807	81.0904
100	86.1142	89.7921	78.3555	81.1722
125	86.9947	89.8494	78.2354	80.8857
150	86.7764	89.4565	77.9807	81.2623
175	86.7528	89.7921	78.6411	80.853
Average	**86.55393**	**89.82483**	**78.20231**	**81.11085**
Standard Dev	**0.33523**	**0.165965**	**0.300583**	**0.326268**

Part Algorithm:



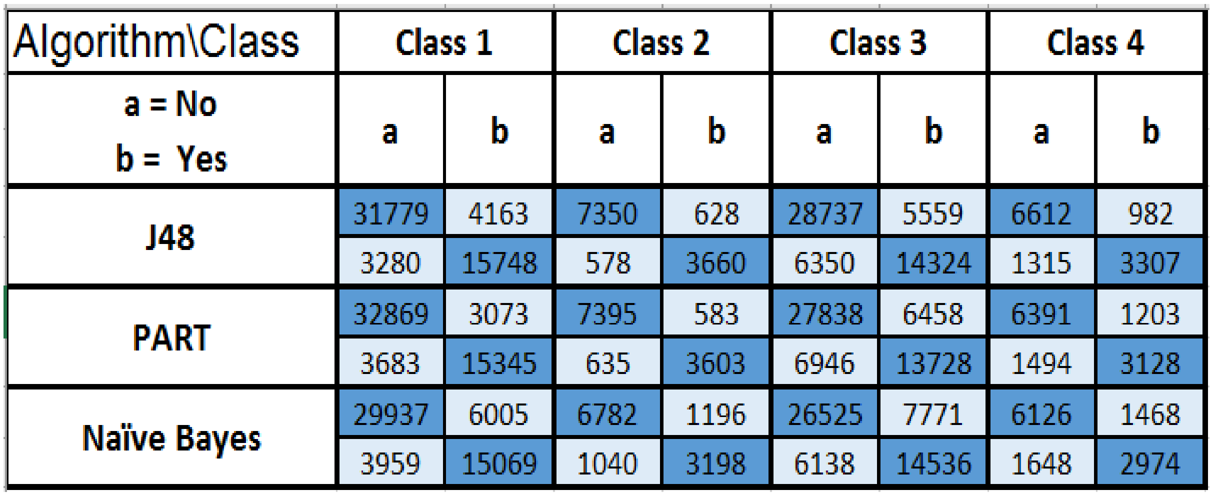
Seed Values	C1	C2	C3	C4
5	87.7097	90.0295	75.6158	77.9224
10	87.6132	90.0704	75.734	78.7492
15	87.9061	90.2177	75.5885	78.7656
20	87.435	90.6762	75.1992	78.4217
25	86.8037	90.2996	73.602	78.2498
50	87.8279	90.2914	74.7753	78.3317
100	87.4113	90.3569	75.4066	78.9784
125	87.4804	90.5124	74.6298	78.6264
150	87.4259	89.9476	74.7644	78.7901
175	86.0433	90.0377	74.9482	78.6182
Average	**87.36565**	**90.24394**	**75.02638**	**78.54535**
Standard Dev	**0.555128**	**0.231941**	**0.63792**	**0.31371**

CONSOLIDATED RESULTS

Algorithms\Factor Cells	C1	C2	C3	C4
Naïve Bayes	81.69037 %	82.21676 %	74.69219 %	75.33154 %
J48	86.55393 %	89.82483 %	78.20231 %	81.11085 %
PART	87.36565 %	90.24394 %	75.02638 %	78.54535 %

The above table shows the consolidated results for each algorithm. The values displayed are the average values for ten runs that were compiled for four scenarios in the experimental design.

Confusion Matrix

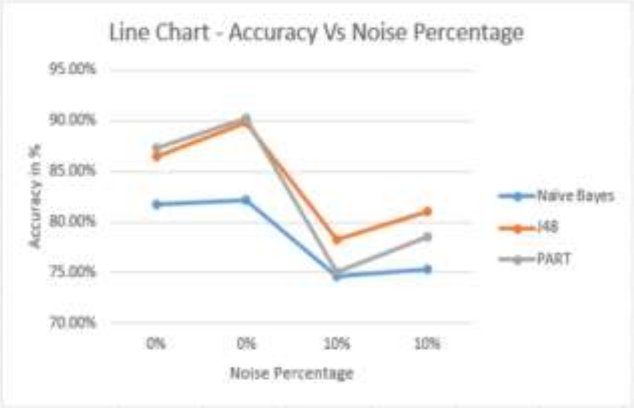


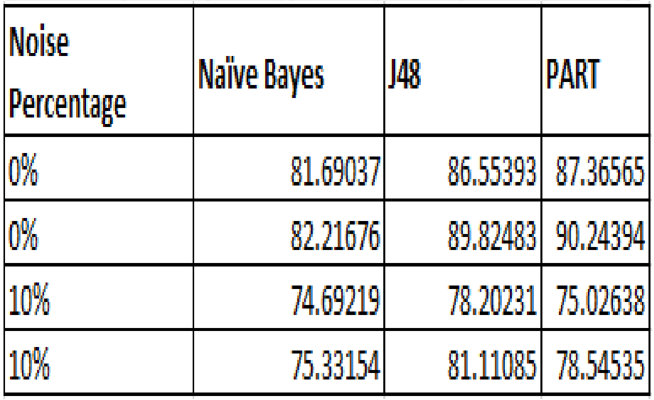
Algorithm\Class	Class 1		Class 2		Class 3		Class 4	
a = No b = Yes	a	b	a	b	a	b	a	b
J48	31779	4163	7350	628	28737	5559	6612	982
	3280	15748	578	3660	6350	14324	1315	3307
PART	32869	3073	7395	583	27838	6458	6391	1203
	3683	15345	635	3603	6946	13728	1494	3128
Naïve Bayes	29937	6005	6782	1196	26525	7771	6126	1468
	3959	15069	1040	3198	6138	14536	1648	2974

Confusion matrix represents the number of correctly classified instances and the wrongly classified instances for each algorithm that was used for this project.

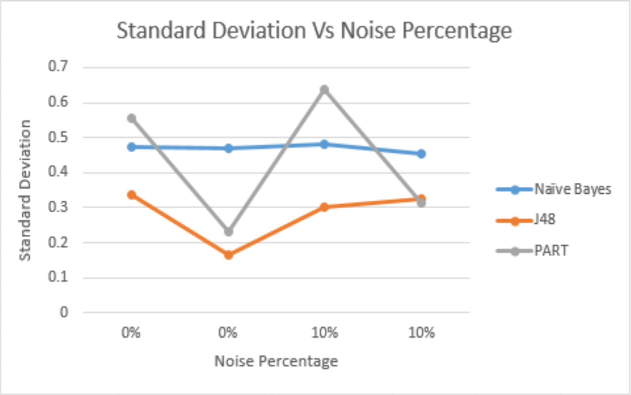
RELATIVE PERFORMANCE OF ALGORITHMS

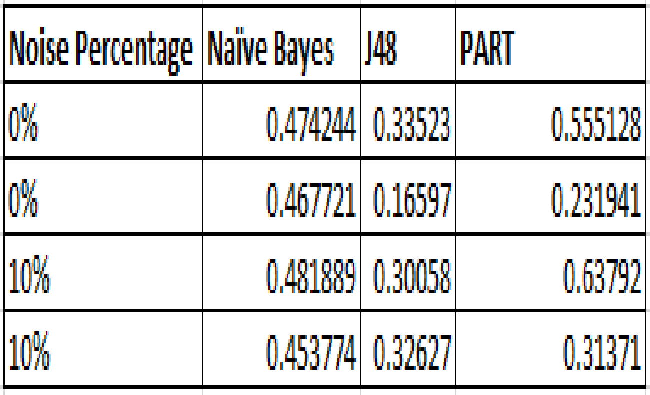
Performance of Classifiers under Noise





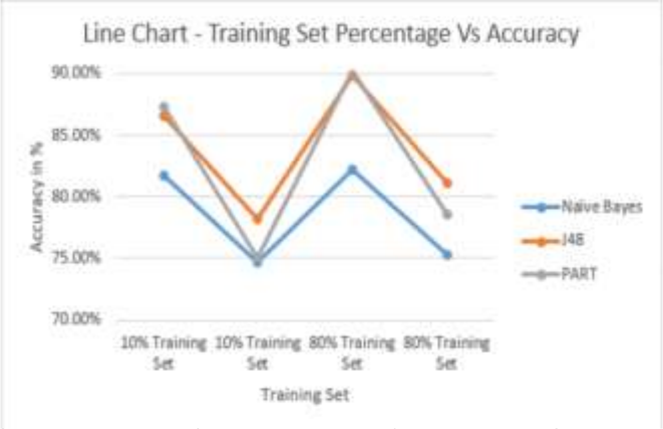
Noise Percentage	Naïve Bayes	J48	PART
0%	81.69037	86.55393	87.36565
0%	82.21676	89.82483	90.24394
10%	74.69219	78.20231	75.02638
10%	75.33154	81.11085	78.54535

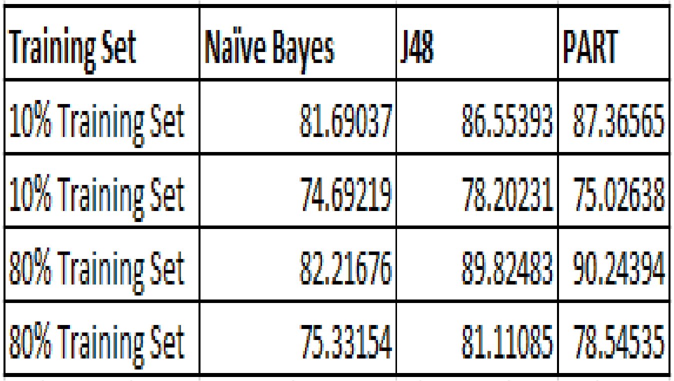




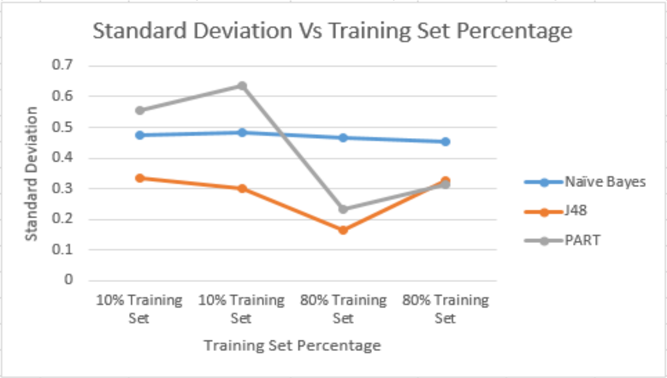
Noise Percentage	Naïve Bayes	J48	PART
0%	0.474244	0.33523	0.555128
0%	0.467721	0.16597	0.231941
10%	0.481889	0.30058	0.63792
10%	0.453774	0.32627	0.31371

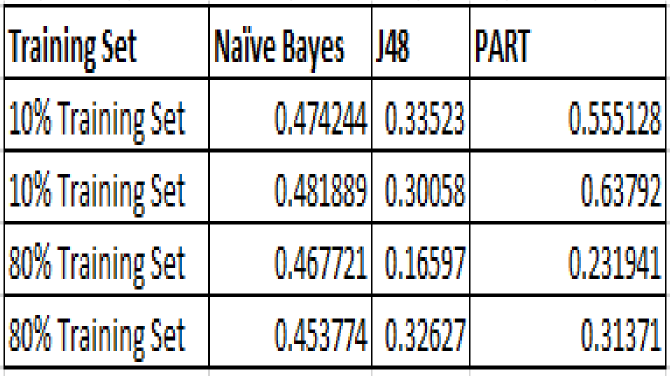
Performance of classifiers under varied training set split





Training Set	Naïve Bayes	J48	PART
10% Training Set	81.69037	86.55393	87.36565
10% Training Set	74.69219	78.20231	75.02638
80% Training Set	82.21676	89.82483	90.24394
80% Training Set	75.33154	81.11085	78.54535





Training Set	Naïve Bayes	J48	PART
10% Training Set	0.474244	0.33523	0.555128
10% Training Set	0.481889	0.30058	0.63792
80% Training Set	0.467721	0.16597	0.231941
80% Training Set	0.453774	0.32627	0.31371

FALSE PREDICTORS

False predictors are values that misdirects the working logic of any algorithm. They sometimes tend to increase the accuracy rate of the working of the algorithm, but are misleading. Such attributes are determined to be false predictors.

- **Last contact duration**–This nominal attribute gives information about the duration of the call happened between the bank and the customer. Ideally predicting the potential customers must have been done prior to making the sales call.
- **Last contact day**– This nominal attribute gives information about the day of the call happened between the bank and the customer. As discussed above, in an ideal scenario, predicting the potential customers must have been done prior to making the sales call. Hence this attribute is a false predictor.
- **Last contact month**– This nominal attribute gives information about the month of the call in a year happened between the bank and the customer. As discussed above, in an ideal scenario, predicting the potential customers must have been done prior to making the sales call. Hence this attribute is a false predictor.
- **Current Campaigns** – This nominal attribute gives information about current campaigns happening in the bank for the sales. But in data mining, usually prediction is done based on past data hence having previous campaign data becomes relevant but current campaign becomes a false predictor.

Testing the dataset without False Predictors:

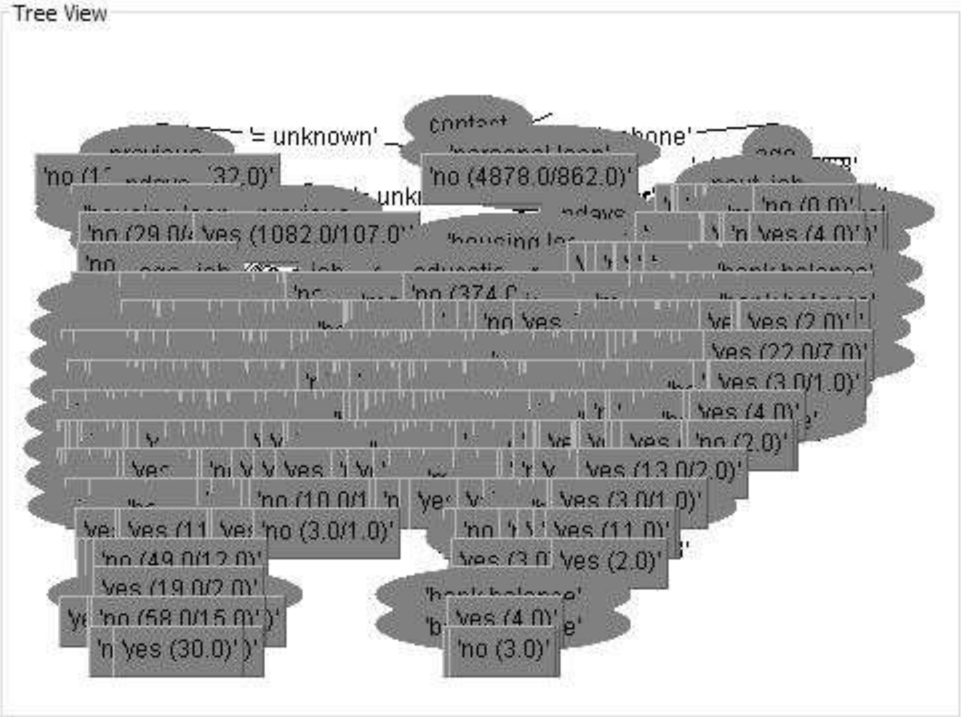
The selected three algorithms are again applied on the dataset using cross validation method after removing the false predictors. The following results are obtained.

Algorithm	0% Noise and 10 folds	10% Noise and 10 folds
Naïve Bayes	73.0263	68.0458
J48	81.129	73.4454
PART	85.1289	73.9563

We are comparing the above results with C2(0% noise & 80% training set) and C4(10% noise & 80% training set). Because C2 and C4 have more percentage in training set (80% instead of 10%). So, the accuracies become more relevant for comparison with current results.

On comparison, it is evident that none of the values in the above table is higher from the earlier results. Still ideally data mining must be done without any false predictors. Therefore, the above-mentioned values are correct accuracies.

Tree Visualization for J48 algorithm:



Confusion Matrix after removing false predictor:

Class/Algorithm	Class 1		Class 2	

a = No
b = Yes
	a	b	a	b

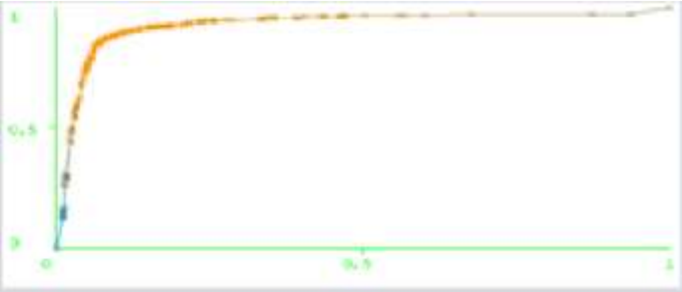
J48	36024	3898	32909	5137
	7628	13528	11082	11950
PART	37352	2570	32862	5184
	6513	14643	10723	12309
Naïve Bayes	32848	7074	30016	8030
	9401	11755	11487	11545

Class 1- 0% Noise & 10 folds

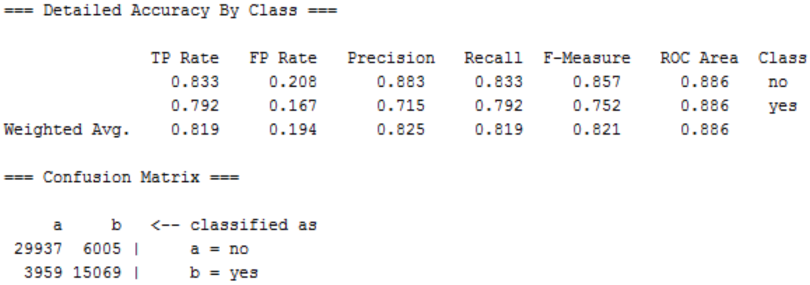
Class 2- 10% Noise & 10 folds.

Receiver Operating Characteristics (ROC) Curve

Receiver Operating Characteristic (ROC) curve is plotted between TPR and FPR. ROC curve plots true positive on the y-axis against false positive on the x-axis. The area covered between the diagonal (threshold line) and curve is AUC (Area Under Curve). The points plotted between TPR and FPR falling in this region determine the accuracy of the algorithm.



<u>Inferences from algorithm runs:</u>



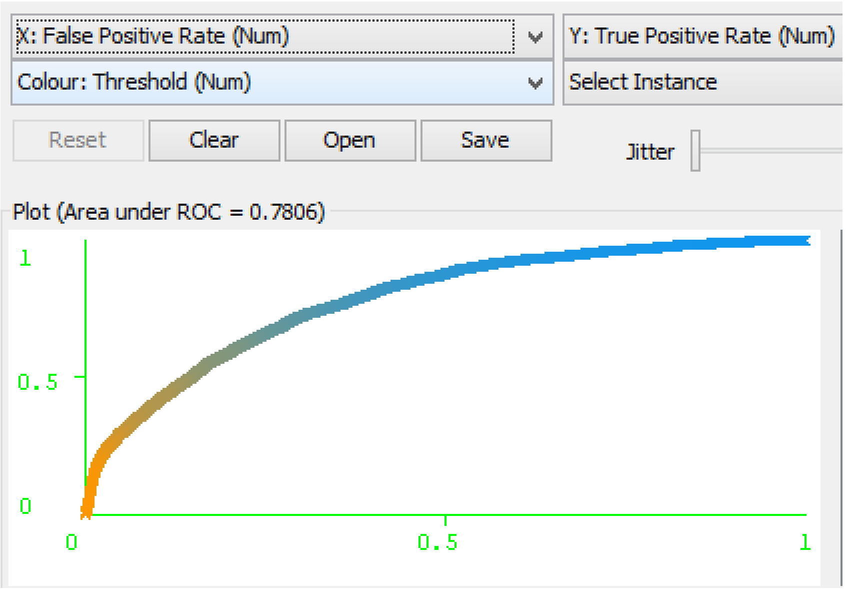
```
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.833     0.208     0.883      0.833     0.857      0.886    no
                 0.792     0.167     0.715      0.792     0.752      0.886    yes
Weighted Avg.    0.819     0.194     0.825      0.819     0.821      0.886

=== Confusion Matrix ===

     a     b   <-- classified as
 29937  6005 |     a = no
  3959 15069 |     b = yes
```

TPR (True Positive Rate) - How many correct positive results are identified among true positive and false negative instances. It is also known as sensitivity or recall.
FPR (False Positive Rate) - How many incorrect positive results occur among false positive and true negative instances. The false-positive rate is also known as the fall-out.

The below given ROC curves were plotted for respective algorithms with cross validation testing and the number of folds was 10.

ROC for Naïve Bayes:

0% Noise (Area under curve is **0.7806**)



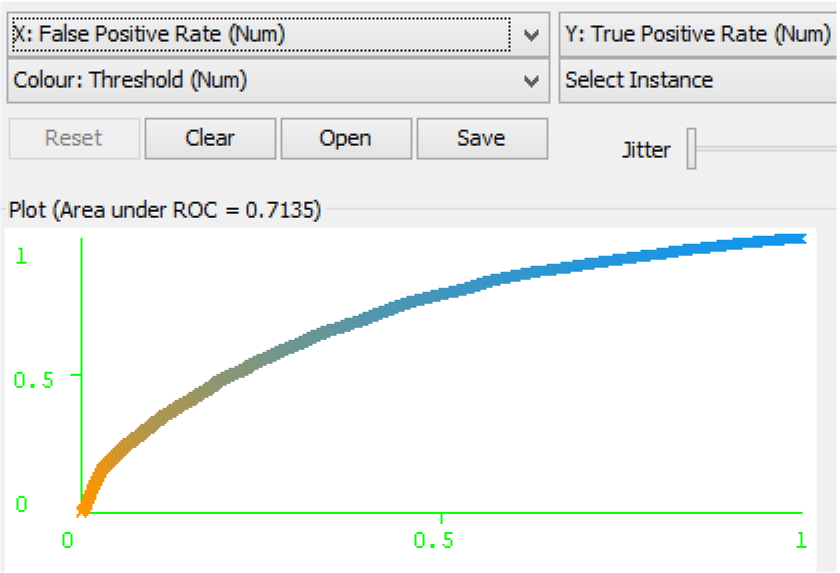
X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.7806)

10% Noise (Area under curve is **0.7135**)



X: False Positive Rate (Num)

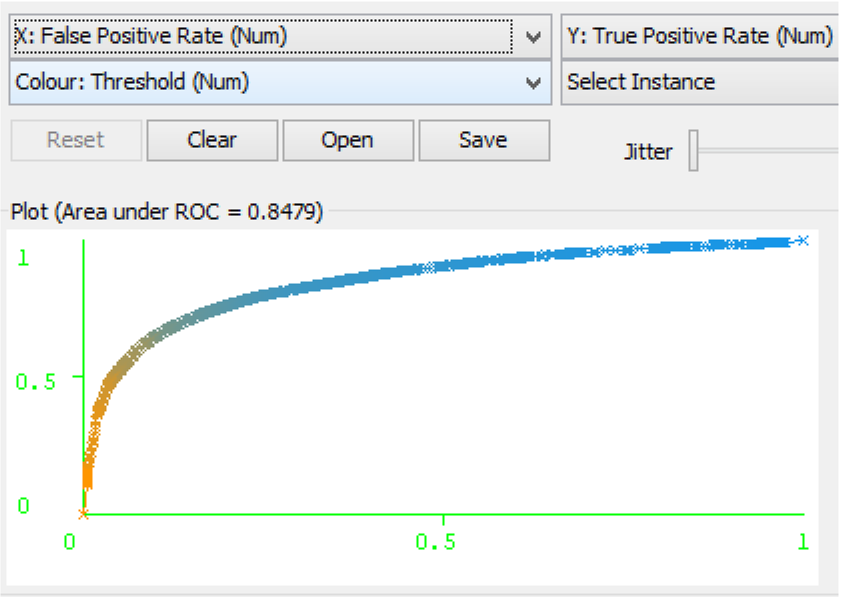
Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.7135)

ROC for J48:

0% Noise (Area under curve is **0.8479**)



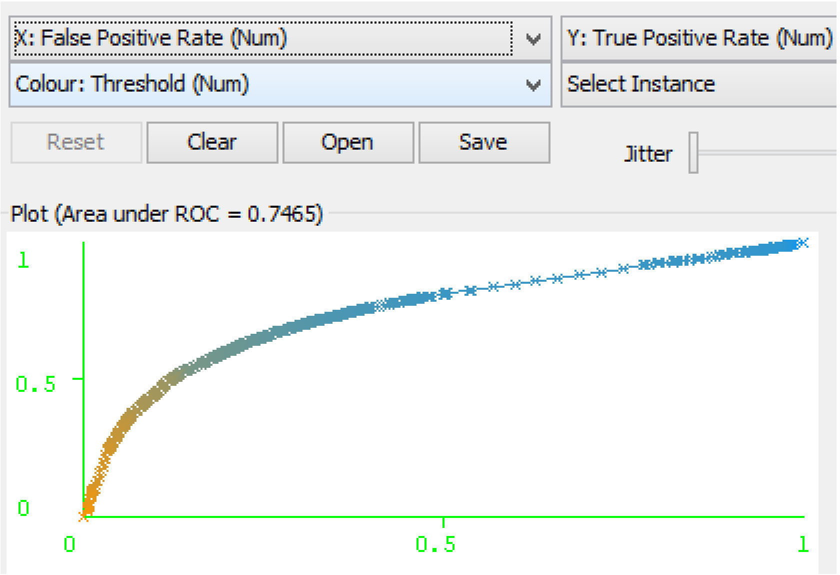
X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.8479)

10% Noise (Area under curve is **0.7465**):



X: False Positive Rate (Num)

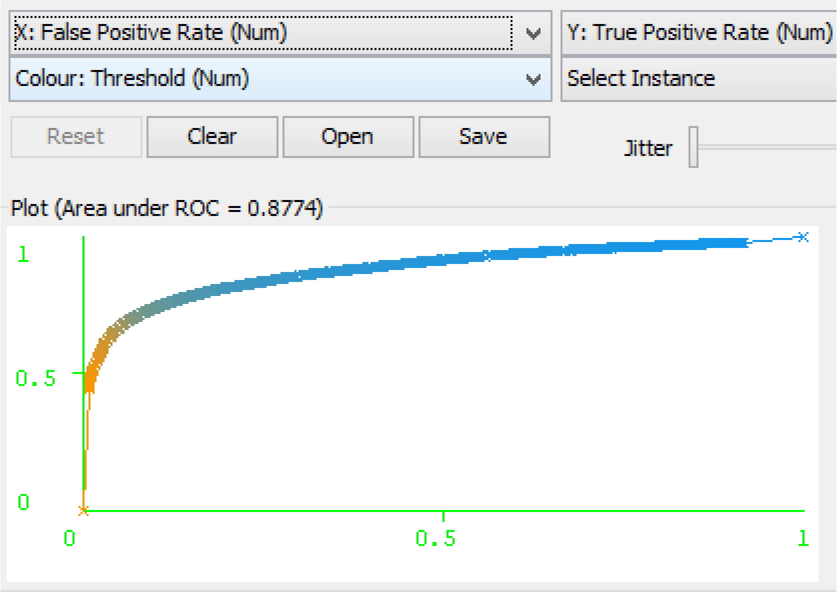
Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.7465)

ROC for PART:

0% Noise (Area under curve is **0.8774**):



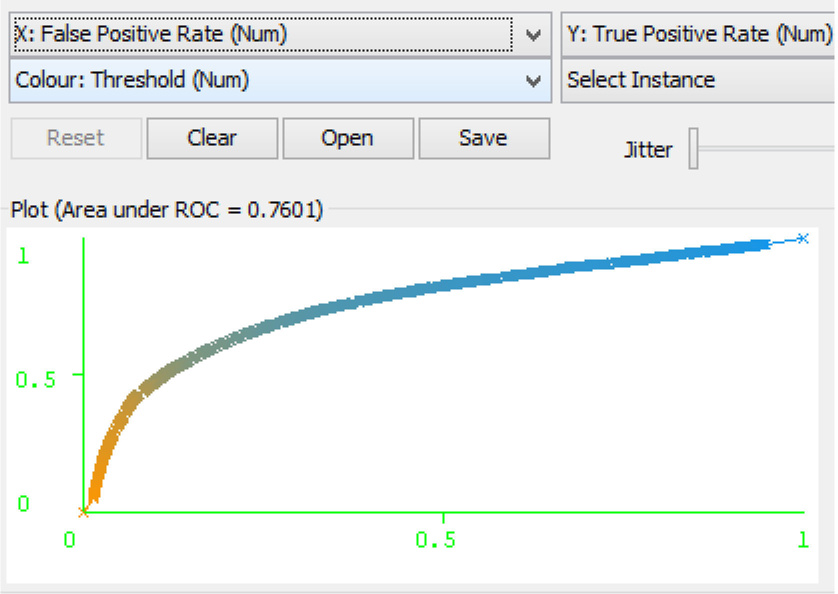
X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.8774)

10 % Noise (Area under curve is **0.7601**)



X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.7601)

CONCLUSION

- After removing the false predictors, the top predictor for J48 algorithm is 'CONTACT' attribute.

- Based on the accuracy obtained from each algorithm and Area Under Curve (AUC), it is possible to conclude that PART algorithm gives the most accuracy for our dataset.

- But Naïve Bayes algorithm performs consistently in the presence of noise.

Algorithm	Correctly Classified Instance	Incorrectly Classified Instance
Naïve Bayes	41561	19517
J48	44859	16219
PART	45171	15907

REFERENCES

[1] Ian Written, Elbe Frank and Mark A. Hall - Data Mining Practice Machine Learning Tools and Techniques

[2] Data pre-processing http://www.cs.ccsu.edu/~markov/ccsu_courses/DataMining-3.html

[3] Data Cleaning and data pre-processing http://www.mimuw.edu.pl/~son/datamining/DM/4-preprocess.pdf

[4] Data Pre-Processing http://searchsqlserver.techtarget.com/definition/data-preprocessing

[5] Algorithms Used http://www.d.umn.edu/~padhy005/Chapter5.html

http://www.ec.tuwien.ac.at/~dieter/research/publications/sac2006.pdf

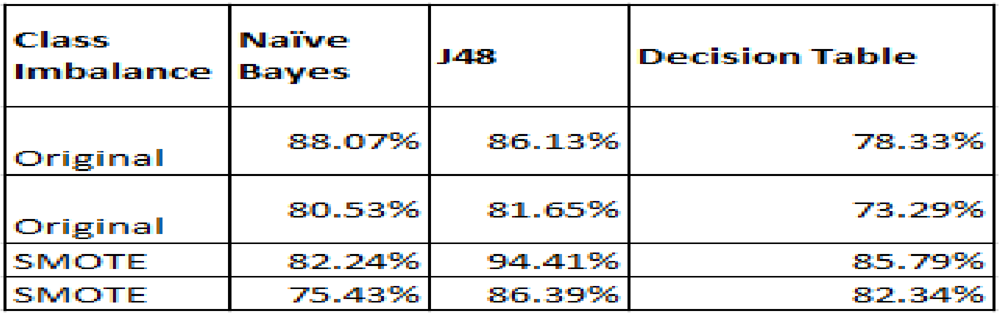
APPENDIX

Since we used SMOTE option over our dataset, we tried with three algorithms (J48, Decision Table, Naïve Bayes) over class imbalance factor and noise.

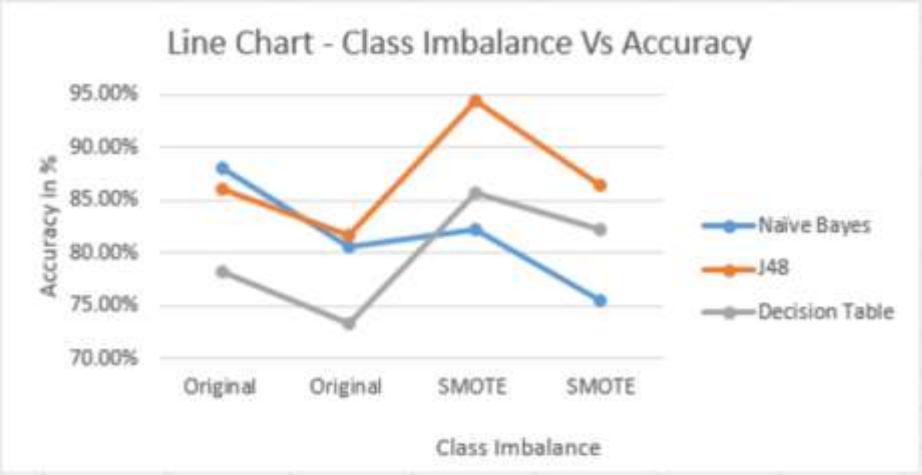
F1- Class Imbalance, F2-Noise
F11- Original, F12- SMOTE, F21- 0% Noise, F22- 10% noise
C1- 0% noise in original dataset
C2- 0% noise in SMOTE dataset
C3- 10% noise in original dataset
C4- 10% noise in SMOTE dataset

Algorithms\Factor cells	**C1**	**C2**	**C3**	**C4**
Naïve Bayes	88.0693%	82.2424 %	80.5291%	75.4331 %
J48	86.1251%	94.4055 %	81.6505%	86.3928 %
Decision Table	78.3261%	85.787 %	73.2919%	82.3357 %

<u>Class Imbalance Versus Accuracy:</u>

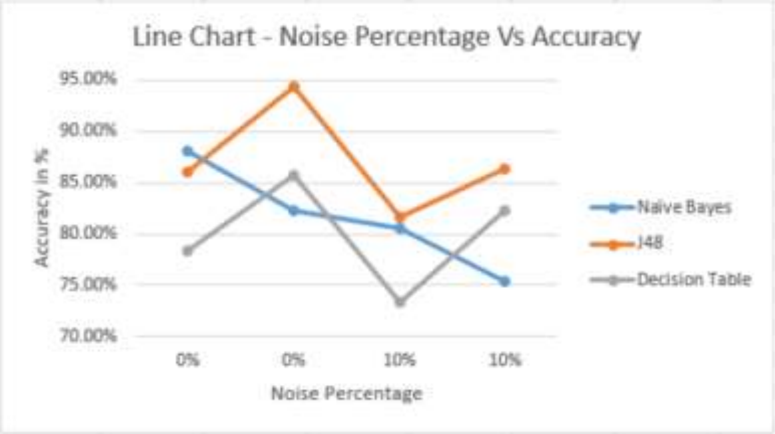


Class Imbalance	Naïve Bayes	J48	Decision Table
Original	88.07%	86.13%	78.33%
Original	80.53%	81.65%	73.29%
SMOTE	82.24%	94.41%	85.79%
SMOTE	75.43%	86.39%	82.34%



Noise Percentage Versus Accuracy:

Noise Percentage	Naïve Bayes	J48	Decision Table
0%	88.0693	86.1251	78.3261
0%	82.2424	94.4055	85.787
10%	80.5291	81.6505	73.2919
10%	75.4331	86.3928	82.3357



Noise Vs SMOTE in Weka

- When we add noise to the dataset, the values of only one attribute is changed. For example, the value of the class variable is changed from ‘yes’ to ‘no’. That is why this option is listed below ‘Attribute’ in Filters.
- But SMOTE option adds additional instances where the minority class can be oversampled. Unlike noise, SMOTE simulates records which has minority class. This is why SMOTE is listed below ‘Instance’ in Filters.

Naïve Bayes and SMOTE

Why there is a decrease in accuracy for Naïve Bayes when SMOTE is used on the dataset?

Naïve Bayes assumes that all attributes are independent of each other. Number of ‘yes’ in dataset remains constant. But the number of ‘no’ in class variable has been increased by SMOTE which in turn increases total number of instances. This reduces the probability of ‘yes’ occurrence. So, while predicting, the output has more chance of classified as ‘no’ than before.

Why accuracy increases when training set percentage is increased?

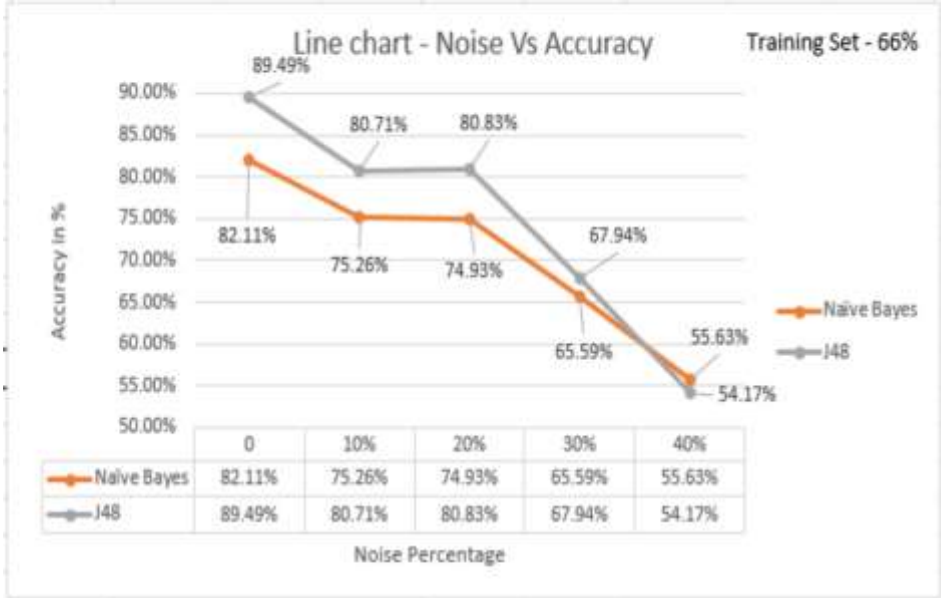
When the training set has very less number of records (e.g. 10%), the algorithms build a model based on these records. These records cover very less scenarios which may occur in the test set. But when the size of the training set is increased (e.g. 80%), the model built on more records will cover more scenarios. So the algorithm results in more accuracy when the size of the training set is more. Performance of the algorithms in the presence of Noise.

Why J48 and PART have low accuracy in the presence of noise?

Naïve Bayes algorithm works based on probability. So the presence of noise has less effect on accuracy. Whereas J48 and PART algorithms work based on rules created by decision trees. These algorithms generate rules which covers the noise while building a model using training set. These rules built predict the wrong output in the test set which results in very less accuracy in the presence of noise.

Performance of Naïve Bayes in presence of noise

This graph shows that when noise percentage was increased, the performance of both Naïve Bayes and J48 decrease. But the accuracy for Naïve Bayes dropped from 82.11% to 55.63% which is comparatively better than J48 (dropped from 89.49% to 54.17%).



	0	10%	20%	30%	40%
Naive Bayes	82.11%	75.26%	74.93%	65.59%	55.63%
J48	89.49%	80.71%	80.83%	67.94%	54.17%