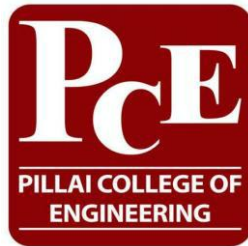


DATA MINING AND BUSINESS INTELLIGENCE

MINI PROJECT ON QUANTITATIVE BANKRUPTCY DATASET



Pillai's College of Engineering

*Dr. K. M. Vasudevan Pillai Campus, Plot No. 10, Sector 16, New Panvel, Navi Mumbai, Maharashtra
410206*

Submitted by

Kaustubh Vaidya 62

Vedant Patil 16

TABLE OF CONTENTS:

1.INTRODUCTION	2
1.1 Business Intelligence:	3
1.2 BI Tools:	4
1.3 Data set:	6
2. PROBLEM STATEMENT	8
2.1 Purpose of Dataset:	8
2.2 Dataset Analysis:	8
3.TECHNIQUE:	9
3.1 Pre-Processing	9
3.2 Classification	11
4. ALGORITHM	12
4.1 Explanation:	12
4.2 Snapshots	16
5. INTERPRET AND VISUALIZE	20
5.1 Explanation :	20
5.2 Visualization Technique Snapshots:	20
6.RESULT	21

1.INTRODUCTION

1.1 Business Intelligence:

Business intelligence (BI) is the use of computing technologies for the identification, discovery and analysis of business data - like sales revenue, products, costs and incomes. Business intelligence (BI) is an umbrella term that includes the applications, infrastructure and tools, and best practices that enable access to and analysis of information to improve and optimize decisions and performance.

BI technologies provide historical, current and predictive views of business operations. Common functions of business intelligence technologies are reporting, online analytical processing, analytics, data mining, process mining, complex event processing, business performance management, bench marking, text mining, predictive analytics and prescriptive analytics.

It is a technology infrastructure for gaining maximum information from available data for the purpose of improving business processes. Typical BI infrastructure components are as follows: software solution for gathering, cleansing, integrating, analyzing and sharing data. Business Intelligence produces analysis and provides believable information to help making effective and high quality business decisions.

The most common kinds of Business Intelligence systems are:

- **EIS** - Executive Information Systems
- **DSS** - Decision Support Systems
- **MIS** - Management Information Systems
- **GIS** - Geographic Information Systems
- **OLAP** - Online Analytical Processing and multidimensional analysis
- **CRM** - Customer Relationship Management

Business Intelligence systems based on Data Warehouse technology. A Data Warehouse(DW) gathers information from a wide range of company's operational systems, Business Intelligence systems based on it. Data loaded to DW is usually good integrated and cleaned that allows to produce credible information which reflected so called 'one version of the true'.

1.2 BI Tools:

The following is a list of desirable features BI portals in particular:

- Usable: User should easily find what they need in the BI tool.
- Content Rich: The portal is not just a report printing tool, it should contain more functionality such as advice, help, support information and documentation.
- Clean: The portal should be designed so it is easily understandable and not over-complex as to confuse the users
- Current: The portal should be updated regularly.
- Interactive: The portal should be implemented in a way that makes it easy for the user to use its functionality and encourage them to use the portal. Scalability and customization give the user the means to fit the portal to each user.
- Value Oriented: It is important that the user has the feeling that the DW/BI application is a valuable resource that is worth working on.

Business intelligence tools are a type of application software designed to retrieve, analyze, transform and report data for business intelligence. The tools generally read data that have been previously stored, often, though not necessarily, in a data warehouse or data mart.

The key general categories of business intelligence tools are:

1. Spreadsheets
2. Reporting and querying software: tools that extract, sort, summarize, and present selected data
3. Online Analytical Processing
4. Digital dashboards
5. Data mining
6. Process Visualization
7. Data warehousing
8. Local information systems

The data mining tool used in this particular dataset mining in RapidMiner.

“RapidMiner provides data mining and machine learning procedures including: data loading and transformation (ETL), data preprocessing and visualization, modelling, evaluation, and deployment. The data mining processes can be made up of arbitrarily nestable operators, described in XML files and created in RapidMiner graphical user interface (GUI). RapidMiner is written in the Java programming language. It also integrates learning schemes and attribute evaluators of the Weka machine learning environment and statistical modelling schemes of the R-Project.”

As you launched RapidMiner Studio (v. 6.1.1000) you will need to install the Text Mining extension. RapidMiner works with extensions that plug into the core system. The Text Mining extension can be found in RapidMiner Marketplace, which can be accessed from Help > Updates and Extensions (Marketplace).

After restarting the software, we can start working with it. First of all create a New Process. You will see now the main window of RapidMiner Studio, and I will briefly describe the main zones of the working space :

- In blue we have the main toolbar
- In orange we can see all the operators that we can use in our processes
- In green we have the repositories
- In purple we have the main process windows, where we will be able to see process results and progression
- In black we have parameters of each element of our process and help

From here, we will first of all find our operator Process Documents from Files and we will drag it into the Process zone, in the center. At this point we have our operator in our process, and we need to set his parameters. Click on our operator in the main process area, and see which parameters you can set on the right side. First parameter is text directories which we will set right away.

Note : On the right side of your toolbar you can see a four-element menu that allows you to switch between Design and Results (also with F8 and F9 keys) that will be very useful. If your results aren't what you were expecting, or you made a mistake when designing your process, you can easily return from the results to the design area.

- In my case, i have a directory on my Desktop which name is "data"
- In /data/, I have crop.csv.
- I will set up my text directories like suggested in the and give both a different name to be able to show results depending on text directory

In next section we will talk about operators, and we will come back to Process Documents from Files parameters to choose which vector we want RapidMiner to create.

1.3 Data set:

1. Dataset Title: Quantitative Bankruptcy
2. Number of Instances: 250
3. Number of Attributes: 6, each corresponding to Qualitative Parameters in Bankruptcy
4. Attribute Information: (P=Positive, A-Average, N-negative, B-Bankruptcy, NB-Non Bankruptcy)
 1. Industrial Risk: {P,A,N}
 2. Management Risk: {P,A,N}
 3. Financial Flexibility: {P,A,N}
 4. Credibility: {P,A,N}
 5. Competitiveness: {P,A,N}
 6. Operating Risk: {P,A,N}
 7. Class: {B,NB}

Internal Risks:

i. Industry risk (IR) :

Government policies and International agreements,
Cyclicalilty,
Degree of competition,
The price and stability of market supply,
The size and growth of market demand,
The sensitivity to changes in macroeconomic factors,
Domestic and international competitive power,
Product Life Cycle.

ii. Management risk(MR):

Ability and competence of management,
Stability of management,
The relationship between management/ owner,
Human resources management,
Growth process/business performance,
Short and long term business planning,
achievement and feasibility.

iii. Financial Flexibility(FF):

Direct financing,
Indirect financing,
Other financing

iv. Credibility (CR):

Credit history,
reliability of information,
The relationship with financial institutes.

v. Competitiveness (CO):

Market position,
The level of core capacities,
Differentiated strategy,

vi. Operating Risk (OP):

The stability and diversity of procurement,
The stability of transaction,
The efficiency of production,
The prospects for demand for product and service,
Sales diversification,
Sales price and settlement condition,
Collection of A/R,
Effectiveness of sale network.

5. Missing Attribute Values: None

6. Class Distribution: [143 instances For Non-Bankruptcy] [107 instances For Bankruptcy]

Information about the dataset

CLASSTYPE: nominal

7. Class Attribute: Class {B,NB}.

Following is the screenshot of Dataset:

ExampleSet (/Local Repository/data/Qualitative_Bankruptcy.data)							
ExampleSet (249 examples, 0 special attributes, 7 regular attributes)							
Filter (249 / 249 examples): all							
Row No. ↑	Industrial Risk	Management Risk	Financial Flexibility	Credibility	Competitive...	Operating Risk	Class
1	N	N	A	A	A	N	NB
2	A	A	A	A	A	A	NB
3	P	P	P	P	P	P	NB
4	N	N	P	P	P	N	NB
5	A	A	P	P	P	A	NB
6	P	P	A	P	P	P	NB
7	P	P	P	A	A	P	NB
8	P	P	A	P	A	P	NB
9	P	P	A	A	P	P	NB
10	P	P	P	P	A	P	NB
11	P	P	P	A	P	P	NB
12	N	N	A	P	P	N	NB
13	N	N	P	A	A	N	NB
14	N	N	A	P	A	N	NB
15	N	N	A	P	A	N	NB
16	N	N	A	A	P	N	NB
17	N	N	P	P	A	N	NB

2. PROBLEM STATEMENT

2.1 Purpose of Dataset:

Many companies are facing Bankruptcy now-a-days. Many types of risk have striking the companies and qualitative and quantitative measurements for company get reduced if not properly maintained. Given dataset states about the possibility of bankruptcy for the company.

2.2 Dataset Analysis:

This dataset performs the test to find t which takes in account different types of risks such as industrial risk, management risk, operating risk and different properties such as credibility, competitiveness and financial flexibility to check that whether the condition will fall under Bankruptcy (B) or Non-Bankruptcy (NB) which helps companies to take respective steps and improve measurements.

3. TECHNIQUE:

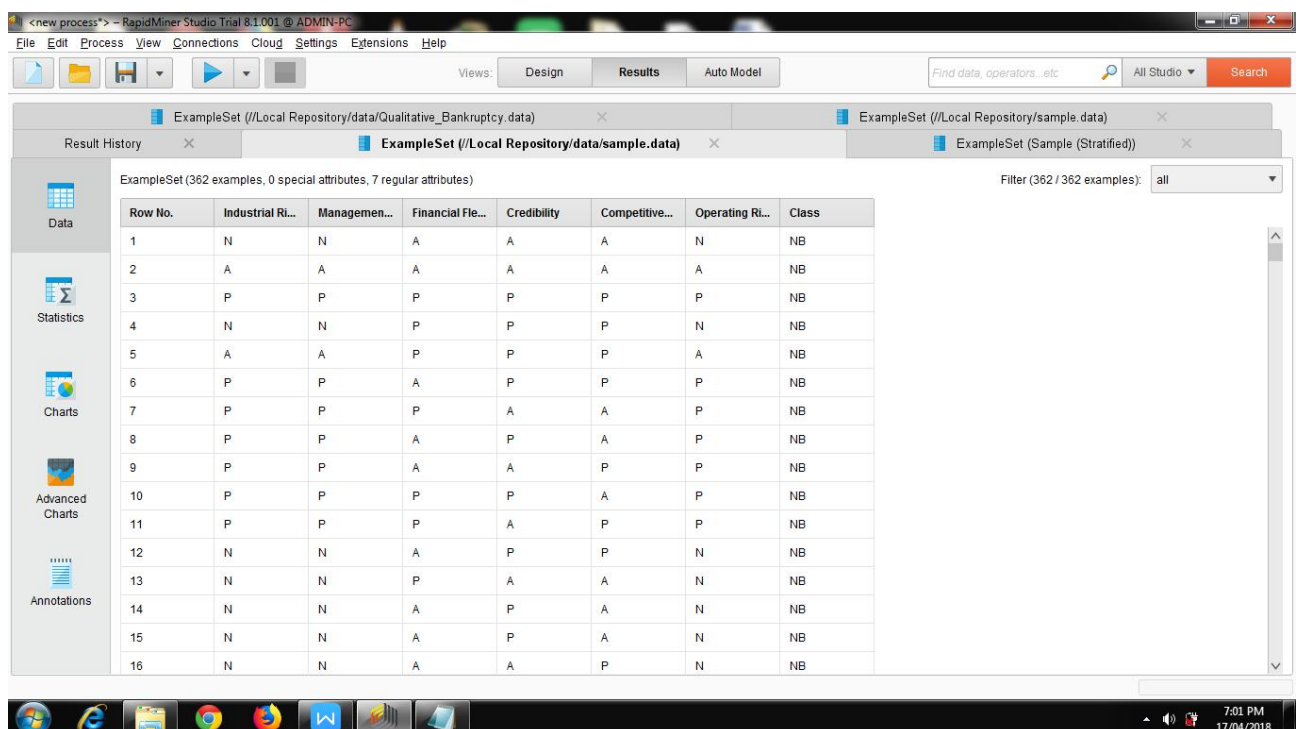
3.1 Pre-Processing:

Sampling allows data scientists, predictive modelers and other data analysts to work with a small, manageable amount of data in order to build and run analytical models more quickly, while still producing accurate findings. Sampling can be particularly useful with data sets that are too large to efficiently analyze in full -- for example, in big data analytics applications. An important consideration, though, is the size of the required data sample. In some cases, a very small sample can tell all of the most important information about a data set. In others, using a larger sample can increase the likelihood of accurately representing the data as a whole, even though the increased size of the sample may impede ease of manipulation and interpretation. Either way, samples are best drawn from data sets that are as large and close to complete as possible.

Stratified sampling strategies

1. *Proportionate allocation* uses a sampling fraction in each of the strata that is proportional to that of the total population. For instance, if the population consists of X total individuals, m of which are male and f female (and where $m + f = X$), then the relative size of the two samples ($x1 = m/X$ males, $x2 = f/X$ females) should reflect this proportion.
2. *Optimum allocation (or disproportionate allocation)* - The sampling fraction of each stratum is proportionate to both the proportion (as above) and the standard deviation of the distribution of the variable. Larger samples are taken in the strata with the greatest variability to generate the least possible overall sampling variance.

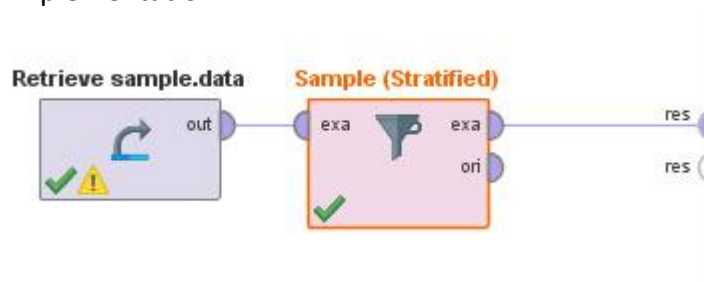
Before Sampling: #362 instances



The screenshot shows the RapidMiner Studio interface. The main window displays a data table titled "ExampleSet (362 examples, 0 special attributes, 7 regular attributes)". The table has 8 columns: Row No., Industrial Ri..., Managemen..., Financial Fle..., Credibility, Competitive..., Operating Ri..., and Class. The data is filtered to show 16 rows. The interface includes a sidebar with icons for Data, Statistics, Charts, Advanced Charts, and Annotations. The top menu bar includes File, Edit, Process, View, Connections, Cloud, Settings, Extensions, and Help. The bottom status bar shows the system clock as 7:01 PM on 17/04/2018.

Row No.	Industrial Ri...	Managemen...	Financial Fle...	Credibility	Competitive...	Operating Ri...	Class
1	N	N	A	A	A	N	NB
2	A	A	A	A	A	A	NB
3	P	P	P	P	P	P	NB
4	N	N	P	P	P	N	NB
5	A	A	P	P	P	A	NB
6	P	P	A	P	P	P	NB
7	P	P	P	A	A	P	NB
8	P	P	A	P	A	P	NB
9	P	P	A	A	P	P	NB
10	P	P	P	P	A	P	NB
11	P	P	P	A	P	P	NB
12	N	N	A	P	P	N	NB
13	N	N	P	A	A	N	NB
14	N	N	A	P	A	N	NB
15	N	N	A	P	A	N	NB
16	N	N	A	A	P	N	NB

Sampling Implementation:



After Sampling: #249 instances

ExampleSet (249 examples, 0 special attributes, 7 regular attributes)

Filter (249 / 249 examples): all

Row No.	Industrial Ri...	Managemen...	Financial Fle...	Credibility	Competitive...	Operating Ri...	Class
1	N	N	A	A	A	N	NB
2	P	P	P	P	P	P	NB
3	N	N	P	P	P	N	NB
4	A	A	P	P	P	A	NB
5	P	P	A	P	P	P	NB
6	P	P	A	P	A	P	NB
7	P	P	A	A	P	P	NB
8	P	P	P	P	A	P	NB
9	N	N	P	A	A	N	NB
10	N	N	A	P	A	N	NB
11	N	N	A	P	A	N	NB
12	N	N	A	A	P	N	NB
13	N	N	P	P	A	N	NB
14	N	N	P	A	P	N	NB
15	A	A	P	A	A	A	NB
16	A	A	P	P	A	A	NB

The data mining technique that is specifically used for this data set is the classification technique.

Data analysis can be used for extracting models describing the important classes is the classification Data mining technique

Data Classification is a two step process:

1. Learning Step:- The training set is analyzed by using a classification algorithm.
2. Classification Step:- It includes a root node, branches and leaf node

3.2 Classification:

The learning and classification steps indicate that it is fast and simple while also having good accuracy for mining the data.

Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify loan applicants as low, medium, or high credit risks.

A classification task begins with a data set in which the class assignments are known. For example, a classification model that predicts credit risk could be developed based on observed data for many loan applicants over a period of time. In addition to the historical credit rating, the data might track employment history, home ownership or rental, years of residence, number and type of investments, and so on. Classifications are discrete and do not imply order. Continuous, floating-point values would indicate a numerical, rather than a categorical, target. A predictive model with a numerical target uses a regression algorithm, not a classification algorithm.

The simplest type of classification problem is binary classification. In binary classification, the target attribute has only two possible values: for example, high credit rating or low credit rating.

Classification models are tested by comparing the predicted values to known target values in a set of test data. The historical data for a classification project is typically divided into two data sets: one for building the model; the other for testing the model.

Scoring a classification model results in class assignments and probabilities for each case. For example, a model that classifies customers as low, medium, or high value would also predict the probability of each classification for each customer.

The classification technique consists of the following methods

1. Decision tree method
2. Naive Bayes method
3. Random Forest method
4. Prune Tree method
5. Neural Network

4. ALGORITHM

4.1 Explanation:

1) Decision Tree:

Decision tree: Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modeling approaches used in statistics, data mining and machine learning. Tree models where the target variable can take a finite set of values are called classification trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically numbers) are called regression trees.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making.

Some techniques, often called ensemble methods, construct more than one decision tree:

1. **Bagging decision trees** : an early ensemble method, builds multiple decision trees by repeatedly re-sampling training data with replacement and voting the trees for a consensus prediction.
2. **A Random Forest** classifier uses a number of decision trees, in order to improve the classification rate.
3. **Boosted Trees** can be used for regression-type and classification-type problems.
4. **Rotation Forest** in which every decision tree is trained by first applying principal component analysis (PCA) on a random subset of the input features.

2) Naive Bayes:

A class' prior may be calculated by assuming equiprobable classes (i.e., $priors = 1 / (\text{number of classes})$), or by calculating an estimate for the class probability from the training set (i.e. $(\text{prior for a given class}) = (\text{number of samples in the class}) / (\text{total number of samples})$). To estimate the parameters for a feature distribution, one must assume a distribution or generate non parametric models for the features from the training set.

The assumptions on distribution of features are called the event model of the Naive Bayes classifier. For discrete features like the ones encountered in document classification (include spam filtering), multi nominal and Bernoulli distributions are popular. These assumptions lead to two distinct models, which are often confused.

Supervised Learning

In Supervised Learning, we are given a set of example pairs (x, y) , $x \in X$, $y \in Y$ and the aim is to find a function that matches the examples. In other words, we wish to infer the mapping implied by the data; the cost function is related to the mismatch between our mapping and the data and it implicitly contains prior knowledge about the problem domain.

A commonly used cost is the mean-squared error, which tries to minimize the average squared error between the network's output, $f(x)$, and the target value y over all the example pairs. When one tries to minimize this cost using gradient descent for the class of neural networks called multilayer perceptions (MLP), one obtains the common and well known back propagation algorithm for training neural networks.

Unsupervised Learning

In Unsupervised Learning, some data x is given and the cost function to be minimized, that can be any function of the data x and the network's output, f .

The cost function is dependent on the task (what we are trying to model) and our a priori assumptions (the implicit properties of our model, its parameters and the observed variables).

As a trivial example, consider the model $f(x)=a$ where a is a constant and the cost $C=E[(x-f(x))^2]$. Minimizing this cost will give us a value of a that is equal to the mean of the data. The cost function can be much more complicated. Its form depends on the application: for example, in compression it could be related to the mutual information between x and $f(x)$, whereas in statistical modeling, it could be related to the posterior probability of the model given the data (note that both of those examples those quantities would be maximized rather than minimized).

3) Random Forest:

Random Forest is a supervised learning algorithm. Like you can already see from its name, it creates a forest and makes it somehow random. The „forest“ it builds, is an ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

- ◆ To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

One big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. I will talk about random forest in classification, since classification is sometimes considered the building block of machine learning.

1) Decision Tree Algorithm:

Algorithm: Generate decision tree. Generate a decision tree from the training tuples of data partition, D .

Input:

Data partition, D , which is a set of training tuples and their associated class labels;
attribute list, the set of candidate attributes;

Attribute selection method, a procedure to determine the splitting criterion that “best” partitions the data tuples into individual classes. This criterion consists of a splitting attribute and, possibly, either a split-point or splitting subset. Output:

A decision tree.

Method:

- (1) create a node N ;
- (2) if tuples in D are all of the same class, C , then
- (3) return N as a leaf node labelled with the class C ;
- (4) if attribute list is empty then

- (5) return N as a leaf node labelled with the majority class in D; // majority voting
- (6) apply Attribute selection method (D, attribute list) to find the “best” splitting criterion;
- (7) label node N with splitting criterion;
- (8) if splitting attribute is discrete-valued and multiway splits allowed then // not restricted to binary trees
- (9) attribute list ← attribute list – splitting attribute; // remove splitting attribute
- (10) for each outcome j of splitting criterion
// partition the tuples and grow subtrees for each partition
- (11) let D_j be the set of data tuples in D satisfying outcome j; // a partition
- (12) if D_j is empty then
- (13) attach a leaf labelled with the majority class in D to node N;
- (14) else attach the node returned by Generate decision tree(D_j , attribute list) to node N; endfor
- (15) return N;

2) Naïve Bayes Algorithm:

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, $X = (x_1, x_2, \dots, x_n)$, depicting n measurements made on the tuple from n attributes, respectively, A_1, A_2, \dots, A_n .

2. Suppose that there are m classes, C_1, C_2, \dots, C_m . 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 naïve Bayesian classifier predicts that tuple X belongs to the class C_i if and only if $P(C_i|X) > P(C_j|X)$ for $1 \leq j \leq m, j \neq i$. Thus, we maximize $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is called the maximum posteriori hypothesis. By Bayes' theorem, $P(C_i|X) = P(X|C_i)P(C_i)/P(X)$

3. As $P(X)$ is constant for all classes, only $P(X|C_i)P(C_i)$ 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, $P(C_1) = P(C_2) = \dots = P(C_m)$, and we would therefore maximize $P(X|C_i)$.

Otherwise, we maximize $P(X|C_i)P(C_i)$. Note that the class prior probabilities may be estimated by $P(C_i) = |C_i, D|/|D|$, where $|C_i, D|$ is the number of training tuples of class C_i in D. 4. Given data sets with many attributes, it would be extremely computationally expensive to compute $P(X|C_i)$. To reduce computation in evaluating $P(X|C_i)$, the naïve 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, $P(X|C_i) = \prod_{k=1}^n P(x_k|C_i)$

$$P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \dots \times P(x_n|C_i).$$

We can easily estimate the probabilities $P(x_1|C_i), P(x_2|C_i), \dots, P(x_n|C_i)$ from the training tuples. Recall that here x_k refers to the value of attribute A_k for tuple X. For each attribute, we look at whether the attribute is categorical or continuous-valued

For instance, to compute $P(X|C_i)$, we consider the following:

- (a) If A_k is categorical, then $P(x_k|C_i)$ is the number of tuples of class C_i in D having the value x_k for A_k , divided by $|C_i, D|$, the number of tuples of class C_i in D.

- (b) If A_k is continuous-valued, then we need to do a bit more work, but the calculation is pretty straightforward. A continuous-valued attribute is typically assumed to have a Gaussian distribution with a mean μ and standard deviation σ , defined by $g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ so that $P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$. These equations may appear daunting, but hold on! We need to compute μ_{C_i} and

σ_{C_i} , which are the mean (i.e., average) and standard deviation, respectively, of the values of attribute A_k for training tuples of class C_i . We then plug these two quantities into Eq. (8.13), together with x_k , to estimate $P(x_k|C_i)$.

For example, let $X = (35, \$40,000)$, where A_1 and A_2 are the attributes age and income, respectively. Let the class label attribute be buys computer. The associated class label for X is yes (i.e., buys computer = yes). Let's suppose that age has not been discretized and therefore exists as a continuous-valued attribute.

3) Random Forest Algorithm:

1. Randomly select "**k**" features from total "**m**" features. Where **k** << **m**
2. Among the "**k**" features, calculate the node "**d**" using the best split point.
3. Split the node into **daughter nodes** using the **best split**.
4. Repeat **1 to 3** steps until "**l**" number of nodes has been reached.
5. Build forest by repeating steps **1 to 4** for "**n**" number times to create "**n**" **number of trees**.

The beginning of random forest algorithm starts with randomly selecting "**k**" features out of total "**m**" features. In the image, you can observe that we are randomly taking features and observations.

In the next stage, we are using the randomly selected "**k**" features to find the root node by using the best split approach.

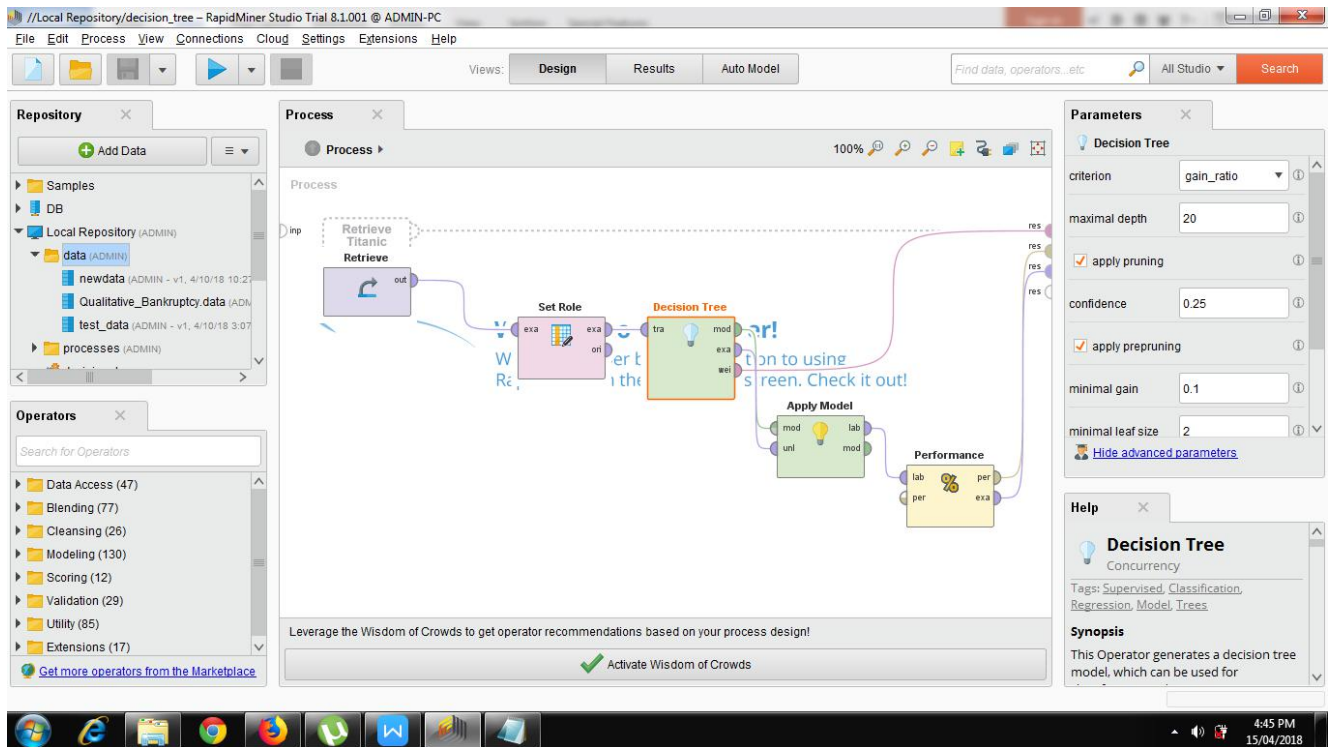
The next stage, We will be calculating the daughter nodes using the same best split approach. Will the first 3 stages until we form the tree with a root node and having the target as the leaf node.

Finally, we repeat 1 to 4 stages to create "**n**" randomly created trees. This randomly created trees forms the random forest.

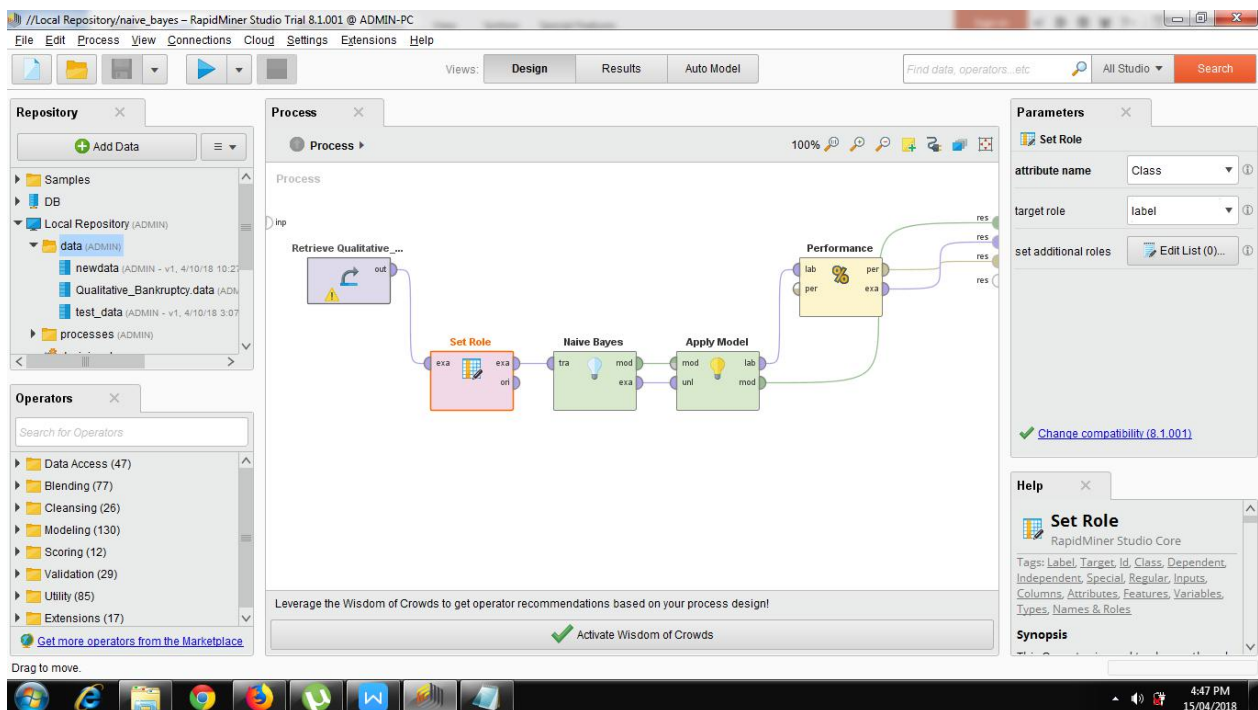
4.2 Snapshots

- Algorithm Implementation:

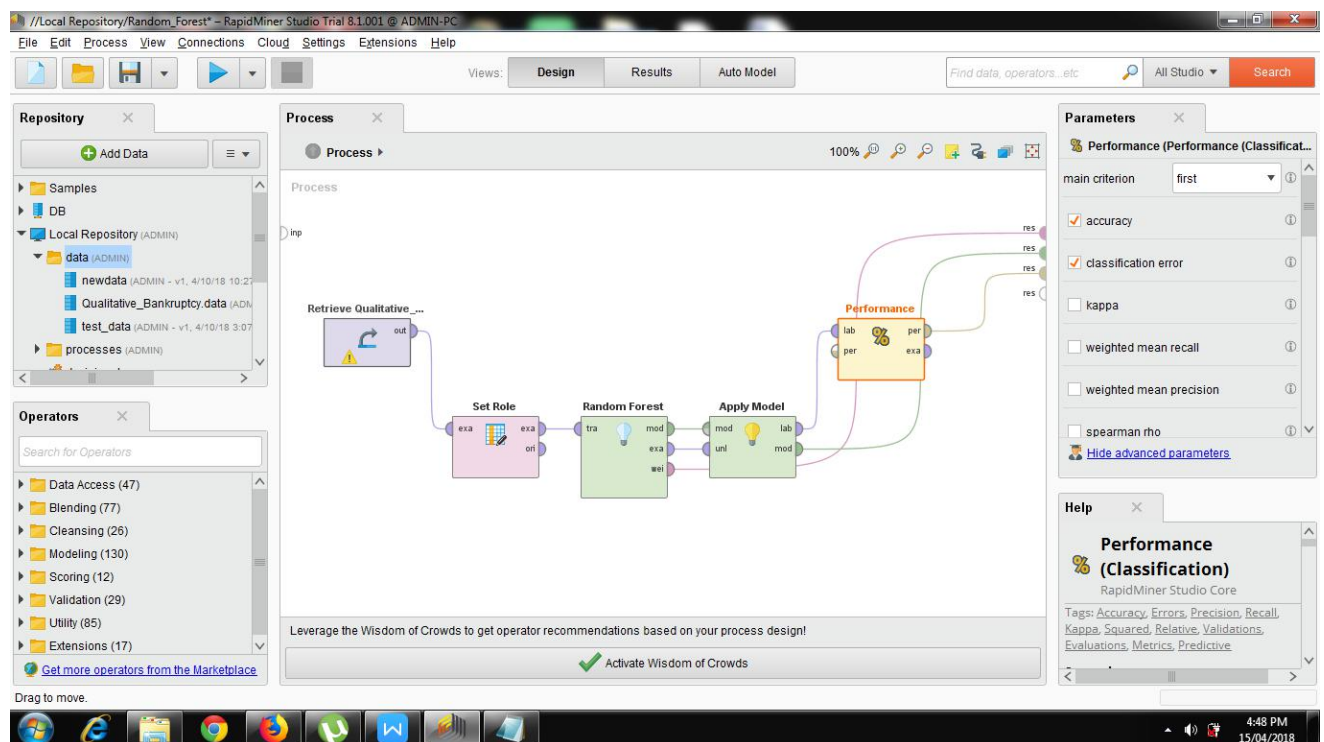
- Decision Tree:



- Naive Bayes:



3. Random Forest:



● Performance Comparison:

Performance of Decision Tree Algorithm:

☒ Table View ☐ Plot View

accuracy: 100.00%

	true NB	true B	class precision
pred. NB	141	0	100.00%
pred. B	0	108	100.00%
class recall	100.00%	100.00%	

Performance of Naive Bayes:

☒ Table View ☐ Plot View

accuracy: 99.60%

	true NB	true B	class precision
pred. NB	141	1	99.30%
pred. B	0	107	100.00%
class recall	100.00%	99.07%	

Performance of Random Forest:

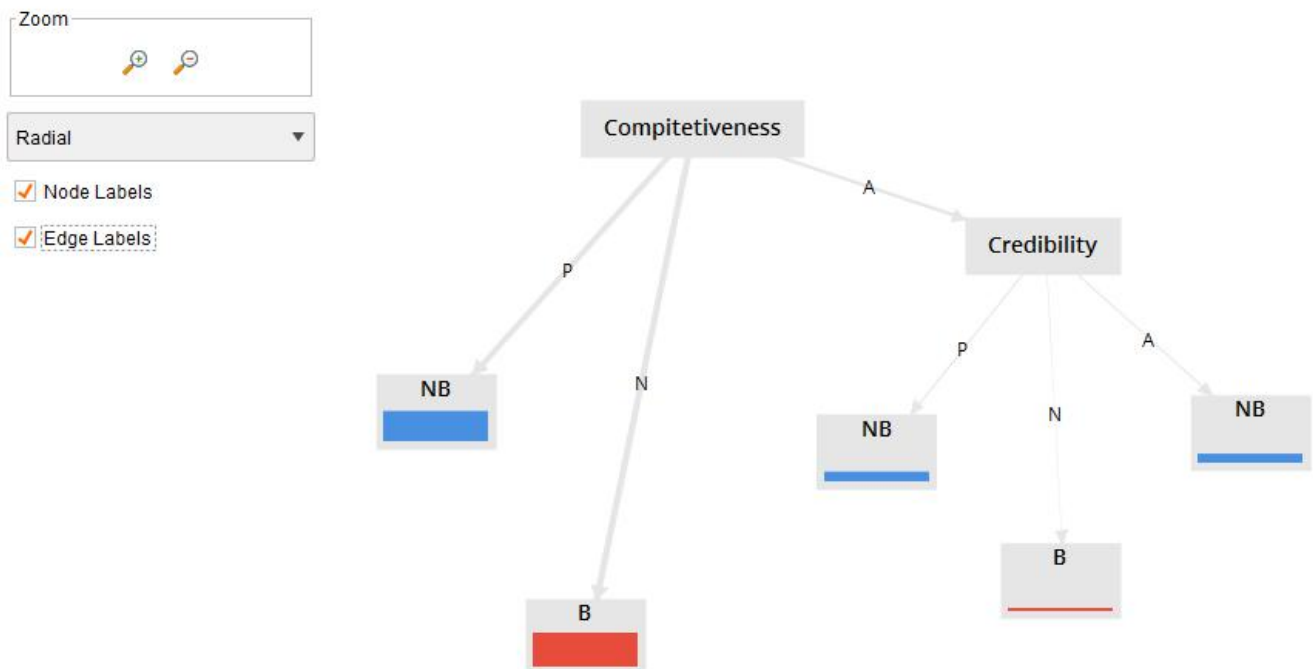
☒ Table View ☐ Plot View

accuracy: 99.60%

	true NB	true B	class precision
pred. NB	141	1	99.30%
pred. B	0	107	100.00%
class recall	100.00%	99.07%	

● Prediction:

Decision Tree Model:



Prediction Implementation:

The screenshot displays the RapidMiner Studio interface in Design view. The process flow is as follows:

- Retrieve Qualitative Data** (Input) connects to **Set Role** (1).
- Set Role** (1) connects to **Decision Tree**.
- Decision Tree** connects to **Explain Predictions**.
- Explain Predictions** outputs to **res** (multiple outputs).
- Retrieve Qualitative Data** also connects to **Set Role** (2).
- Set Role** (2) connects to **Decision Tree**.
- Decision Tree** connects to **Explain Predictions**.
- Explain Predictions** outputs to **res** (multiple outputs).

The **Parameters** panel for **Set Role** is visible on the right, showing:

- attribute name:** Class
- target role:** label
- set additional roles:** Edit List (0)...

The **Help** panel for **Set Role** is also visible, showing tags and a synopsis.

Prediction:

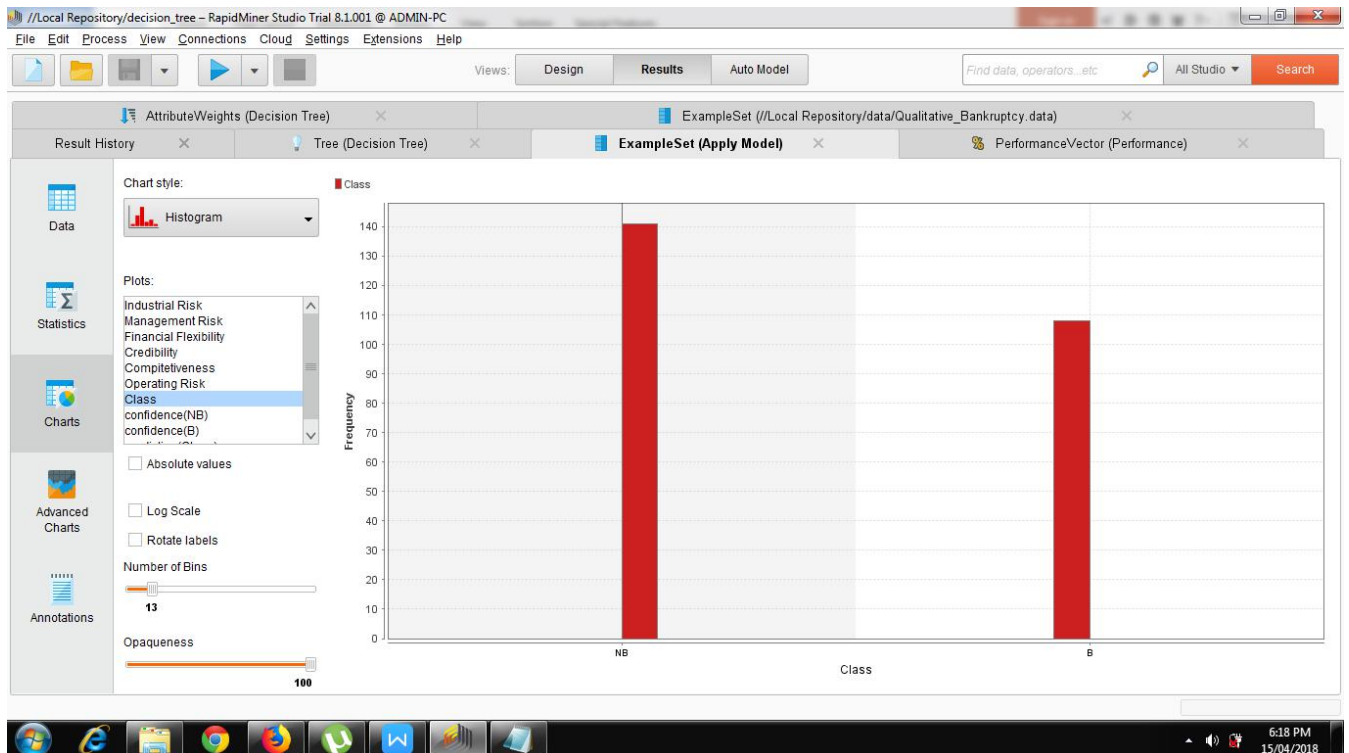
The screenshot displays the RapidMiner Studio interface in Results view. The **ExampleSet (Explain Predictions)** is shown, with the **ExplainPredictionsObject (Explain Predictions)** selected. The results are displayed in a table with 11 columns and 8 rows.

Row No.	Class	prediction(Class)	confidence(...)	confidence(B)	Industrial RI...	Managemen...	Financial Fle...	Credibility	Competitive...	Operating RI...
1	?	NB	1	0	N	P	P	P	P	N
2	?	NB	1	0	P	N	P	P	P	N
3	?	NB	1	0	N	N	A	P	P	P
4	?	NB	1	0	P	N	P	A	A	P
5	?	NB	1	0	N	P	A	P	A	P
6	?	NB	1	0	N	P	A	A	P	N
7	?	B	0	1	A	N	N	N	N	A
8	?	B	0	1	P	N	N	N	N	N

5. INTERPRET AND VISUALIZE

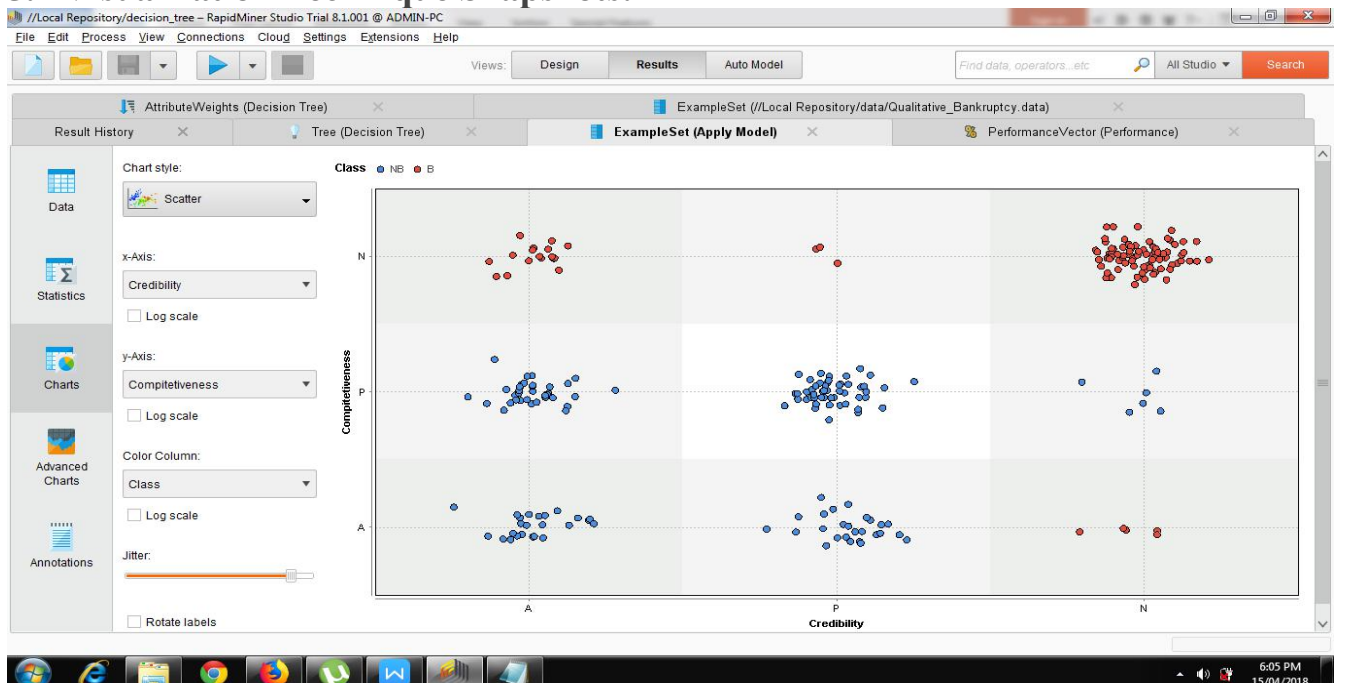
5.1 Explanation :

In the following histogram, Bankruptcy Distribution is shown:



Due to this bankruptcy can be deduced, which helps to know about measurements to handle by companies. Therefore, it becomes possible to help to increase the Quality of company. The particular company can have different type of risks to be faced. So this explains us improvements that can be introduced by different methods.

5.2 Visualization Technique Snapshots:



6.RESULT

Overall Error report

As we performed different classification methods, We found the accuracy and error percentage of this methods for Bankruptcy Dataset. As the dataset was real time, classification performed is on real basis and naturally acceptable. Hence the classification techniques performed well. The error for decision tree is 0%, For Naive Bayes is 0.40% and for Random Forest is 0.40%.

Model Type	Accuracy	Error
Decision Tree	100%	0%
Naive Bayes	99.60%	0.40%
Random Forest	99.60%	0.40%

Therefore, the Decision Tree classification technique is the most compatible data mining technique for the Given data set than Naive Bayes and Random Forest.