**Goal:**

* To outline the key ideas, strategies, and resources for working with data and creating prediction models.
* To clean and divide the dataset according to the given instructions
* To build a Naïve Bayes and random forest models
* Being able to explain the outputs

**Materials**

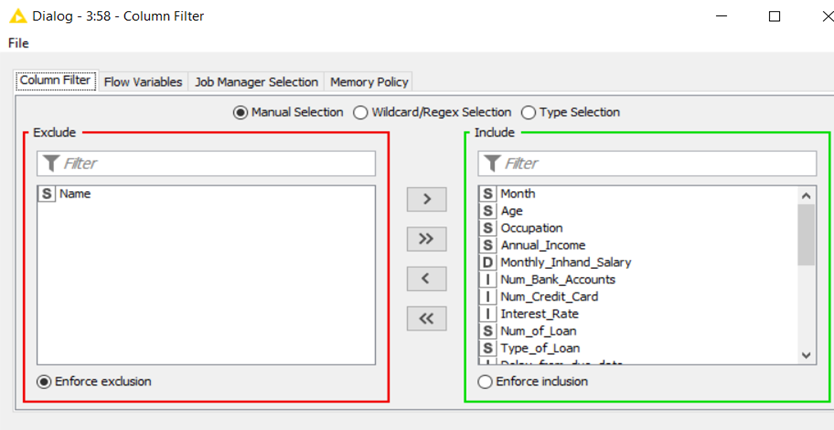
* KNIME Analytics Platform
* Data\_2023.csv

**Method**

* Using different nodes in KNIME
* Naïve Bayes classifier in KNIME
* Random forest classifier in KNIME

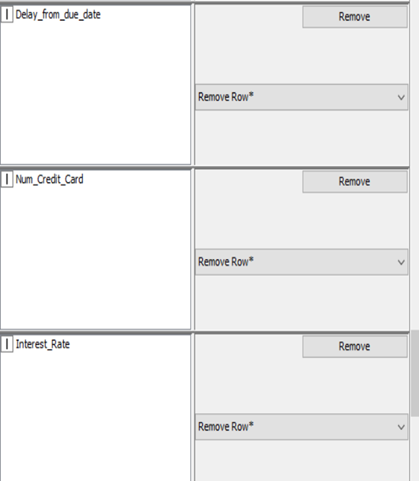
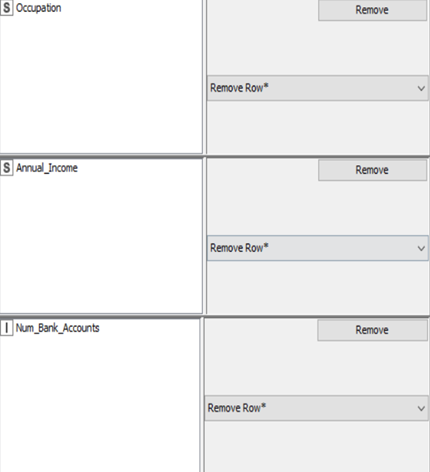
**Procedure and Results:**

1. **DATA CLEANING**
2. I found that there is an attribute called “Name” which is irrelevant to predicting “Credit Score”. When “Name” is removed, it does not affect the final results. Besides, the requirement forces us to employ a limit of 600 distinct nominal values per attribute, however, the attribute "Name" has more than 600 nominal values, which is the primary reason I want to remove it from the data set.



Secondly, I start to remove tuples containing missing values in some attributes such as “Month”, “Age”, “Occupation”, “Annual\_Income”, “Num\_Bank\_Accounts”, “Num\_Credit\_Card”, “Interest\_Rate”, “Num\_of\_Loan”, “Delay\_from\_due\_date”, “Changed \_Credit\_Limit”, “Credit\_Mix”, “Outstanding\_debt”, “Credit\_Utilization\_Ratio”, “Credit\_History\_Age”, “Payment\_of\_Min\_Amount”, “Total\_EMI\_per\_month”, “Amount\_invested\_monthly” and “Payment\_Behavior”.

I use the “Missing value” node for this step. Here are some of them, other attributes are the same.



Additionally, I exclude some tuples with infeasible values from the provided data.

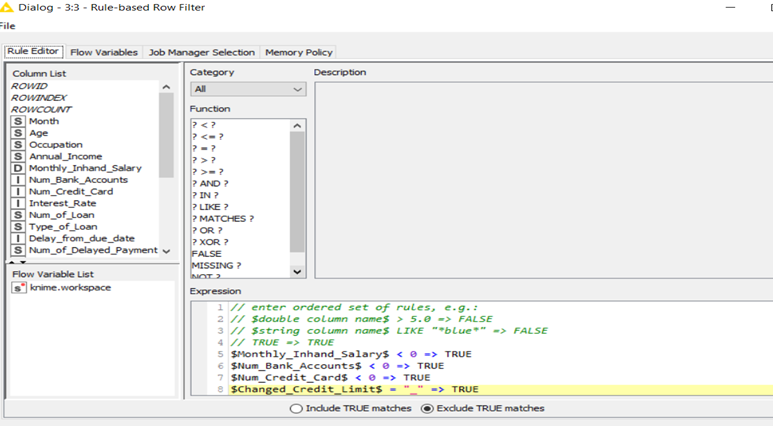
+ “Monthly\_Inhand\_Salary” < 0

+ “Num\_Bank\_Accounts” < 0

+ “Num\_Credit\_Card” < 0

+ “Changed\_Credit\_Limit” contains “\_\_\_”

I use the “Rule-based Row Filter” node for this requirement.



Impossible values will be assigned “true”. It means that those are chosen to be excluded.

1. Next is to eliminate symbols that are non-numerical and convert them into the usual number format, followed by removing tuples which values greater than 120 or lower than or equal to 0 in the “Age” attribute.

|  |  |  |
| --- | --- | --- |
| Sequence | Node | Command |
| 1 | String Manipulation |  |
| 2 | Rule-based Row Filter |  |

I use the “String Manipulation” node and “Rule-based Row Filter” node for this step.

1. Similar to step 3, I delete the non-numerical symbol and convert it to the double format.

I use the “String Manipulation” node for this step.

|  |  |  |
| --- | --- | --- |
| Sequence | Node | Command |
| 1 | String Manipulation |  |

The regular expression used in this case is "\_", which matches any underscore character in the string.

The regexReplace() function replaces all matches of the regular expression in the $Annual\_Income$ string with an empty string(“”).The resulting string (with underscores removed) is then passed to the toDouble() function, which converts the string to a floating-point number

1. I use the “String Manipulation”, “Math Formula”, and “Math Formula (Multi Column) nodes for this step. Basically, the commands are similar to the command in step 4. The noticeable nodes are Math Formula and Math Formula (Multi Column)

Math Formula(Multi Column) explanation: the command abs($$Current\_Column$$) returns the absolute value of the numeric value represented by $$ Current\_Column $$.

Math Formula explanation: The command checks if the value of the variable $Num\_of\_Loan$ is less than zero. If it is, the value 0 is returned. If it is not less than zero, then the original value of $Num\_of\_Loan$ is returned. In other words, this command ensures that the value of $Num\_of\_Loan$ is never negative.

**( The table is under 6. table. Sorry for the inconvenience )**

1. Next, I convert the “Credit\_History\_Age” to the total of months and keep it in the integer format. The final results are stored in a new attribute called “Total\_CHA”.

|  |  |  |  |
| --- | --- | --- | --- |
| Sequence | Node | Command | Main Function |
| 1 | String Manipulation |  | Remove the non-numerical symbol and convert it to integer data type (“Num\_of\_Loan” attribute) |
| 2 | String Manipulation |  |
| 3 | Math Formula |  | Take the absolute values of two attributes “Num\_Bank\_Account” and “Num-Credit\_Card” |

Two “String Manipulation” are quite the same (extracting a two-digit number from a string stored in the $Credit\_History\_Age$ variable) and the difference is that the second one starts where the number is located immediately after the first occurrence of the letter "d".

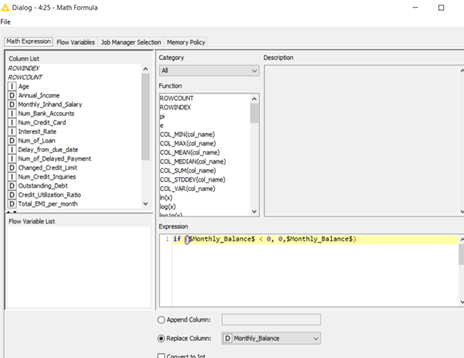
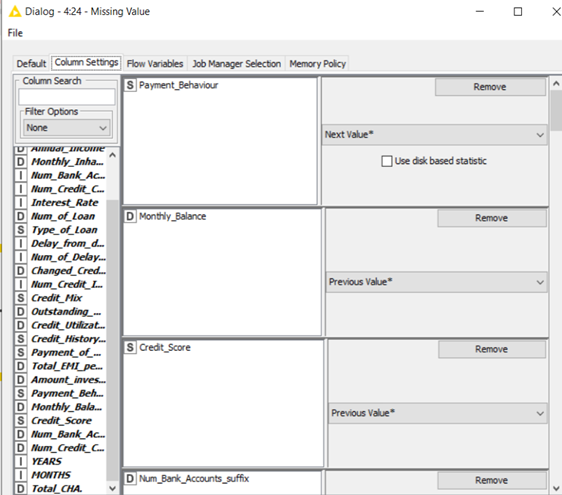
|  |  |  |  |
| --- | --- | --- | --- |
| Sequence | Node | Command for task 5 | Main Function |
| 1 | String Manipulation |  | Convert the “\_\_” to Null (“Occupation” attribute) |
| 2 | String Manipulation |  | Remove the non-numerical symbol and convert it to integer data type (“Num\_of\_Loan” attribute) |
| 3 | Math Formula (Multi Column) |  | Take the absolute values of two attributes “Num\_Bank\_Account” and “Num-Credit\_Card” |
| 4 | Math Formula |  | Set values to 0 if the initial values are lower than 0 (“Num\_of\_Loan”) |
| 5 | String Manipulation |  | Exclude the non-numerical symbol and convert it into integer format (“Num\_of\_Delayed\_payment”) |
| 6 | String Manipulation |  | Set values to “Unknown” if the initial values are “\_” (“Credit\_Mix”) |
| 7 | String Manipulation |  | Exclude the non-numerical symbol and convert it into double format (“Outstanding\_Debt”) |

**To keep the report no more than 10 pages, from here I just insert pictures where is necessary.**

|  |  |  |  |
| --- | --- | --- | --- |
| Sequence | Node | Command | Main Function |
| 1 | String Manipulation | toDouble(regexReplace($Amount\_invested\_monthly$,"\_\_" ,"" )) | Remove the non-numerical symbol in “Amount\_invested\_monthly” and convert it into the double format |
| 2 | String Manipulation |  | Replace the value to “Unknow” if the initial value in “Payment Behaviour” attribute strats with “!@” |
| 3 | String Manipulation | toDouble(regexReplace($Monthly\_Balance$,"\_\_" ,"" )) | Exclude non-numerical symbol in “Monthly Balance” and convert it into the double format |
| 4 | String Manipulation | toDouble(regexReplace($Changed\_Credit\_Limit$,"\_" ,"" )) | Convert “Changed\_Credit\_Limit” into the double format |

1. The seventh task
2. Graphical user interface, text, application

   Description automatically generatedI implement the "Missing Value" node in this task and use the "Next Value" node to fill in any missing values in any string type attributes. To replace missing values in numerical format, I also utilize the "Previous Value" node in the same node.



In addition, if the value of the "Monthly\_Balance" column is negative, I use the "Math Formula" node to replace the value with 0

|  |  |  |  |
| --- | --- | --- | --- |
| Sequence | Node | Command | Main Function |
| 1 | String Manipulation | string($Type\_of\_Loan$) |  |
| 2 | String Manipulation | substr($Type\_of\_Loan$,0,indexOf($Type\_of\_Loan$,"," ) ) | Keep the first part of the original content of the “Type of Loan” attribute has more than one type separated by a comma. |
| 3 | Rule Engine | $Type\_of\_Loan$ LIKE "" => $new\_loan$  TRUE => $Type\_of\_Loan$ | sets the value of the "Type\_of\_Loan" variable to "new\_loan" if it is currently an empty string, otherwise it returns the current value of "Type\_of\_Loan" |

9.

10. For this task, use a “Numeric Biner” node to bind the “Changed\_Credit\_Limit” attribute to the six bins of the range and store the final result in a new column called “Changed\_Credit\_Limit\_binned”.

Graphical user interface

Description automatically generated

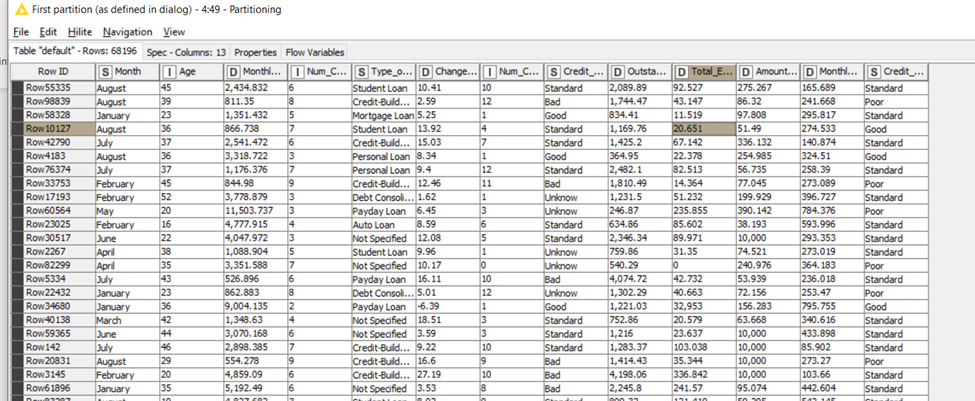
Graphical user interface, application

Description automatically generated11. After using “Column Filter to remove 4 useless attributes including: “YEARS”, “MONTHS”, “Total\_CHA” and “new\_loan”, I use “Feature Selection Loop Start (1:1)” node to select the feature. This node uses the Genetic Algorithm as the trait selection strategy with default population size and the maximum number of generations. Also as a static random seed, I set 3122. After that, I use “Shuffle” node with seed 3122. After this node, I finally use “Partitioning” node to split the data set into training set (75%) and test set (25%).

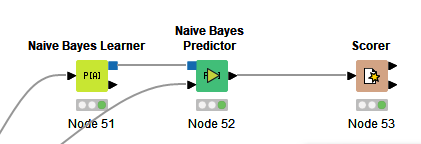
Graphical user interface, application

Description automatically generated

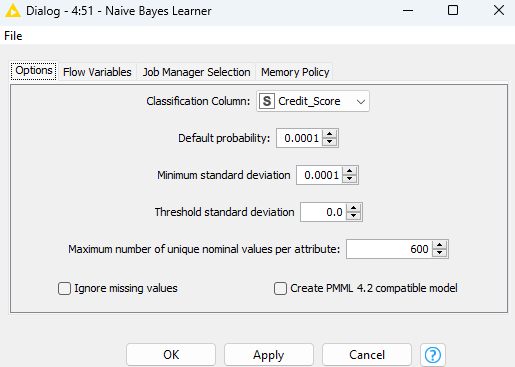
Finally, there are 68196 tuples and 13 attributes in the training data set at the end.



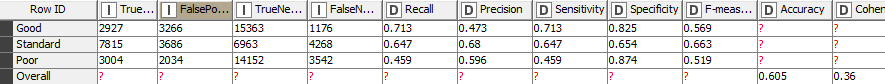
1. **NAÏVE BAYES CLASSIFIER**
2. The screenshot of the Naïve Bayes model

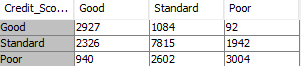
****

1. In the Naive Bayes Learner node, I set the standard probability to 0.0001, the minimum standard deviation to 0.0001, the threshold standard deviation to 0, and the maximum number of unique nominal values ​​per attribute to 600.

****

1. Screenshots of the matrix and the Accuracy statistics of the test result.

****

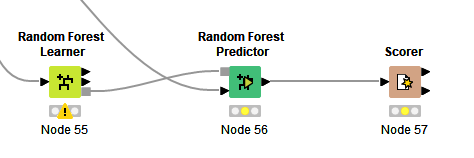
****

The “Good” in “Credit\_Score” has the lowest precision (0.473) of all three categories including “Standard” and “Poor” (0.68 and 0.596 respectively). If the bank wants to decrease the risk of lending money to customers, the “Good” should be the major target, it also means that “Good” should have the highest precision rate. Therefore, the classifier does not achieve this major target.

1. The precision rate should be the measurement we should focus on interpreting the conclusion.

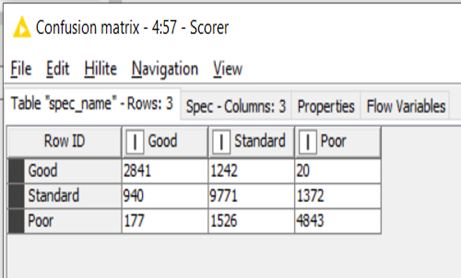
The Recall rate is high. Thus, the rate of missing “Good” cases in reality is low. But the precision rate is low, this model still does not satisfy banks in minimizing the risk of lending money.

1. **RANDOM FOREST CLASSIFIER**
2. The screenshot of the Random Forest model



1. Screenshots of the matrix and the Accuracy statistics of the test result.

Table

Description automatically generated****

3. As I mentioned, the “Good” in “Credit\_Score” should be the major target if the bank wants to minimize the risk of lending money. We should use the precision rate for the measurement. Quickly comparing the result between Naïve Bayes and the Random Forest model, it is apparent that the Random Forest classifier gives a higher Precision rate (0.718) than that of Naïve Bayes.

4. “Standard” is the class that is performed the best by the classifier. Besides the precision rate, I believe the recall rate also should be looked at as we are finding the answer.