Cover sheet for submission of work for assessment

SWINBURNE UNIVERSITY OF TECHNOLOGY

UNIT DETAILS

Unit name	Data Science Principles			Class day/time	Wednesday
Unit code	COS10022	Assignment no.	2	Due date	26th March
Name of lecturer/teacher	Kim Dung Pham				
Tutor/marker's name					

Office use only

Faculty or school date stamp

STUDENT(S)

	Family Name(s)	Given Name(s)	Student ID Number(s)
(1)	Luong	Trac Duc Anh	103488117
(2)			
(3)			
(4)			
(5)			
(6)			

DECLARATION AND STATEMENT OF AUTHORSHIP

1. I/we have not impersonated, or allowed myself/ourselves to be impersonated by any person for the purposes of this assessment.
2. This assessment is my/our original work and no part of it has been copied from any other source except where due acknowledgement is made.
3. No part of this assessment has been written for me/us by any other person except where such collaboration has been authorised by the lecturer/teacher concerned.
4. I/we have not previously submitted this work for this or any other course/unit.
5. I/we give permission for my/our assessment response to be reproduced, communicated, compared and archived for plagiarism detection, benchmarking or educational purposes.

I/we understand that:

6. Plagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offence that may lead to exclusion from the University. Plagiarised material can be drawn from, and presented in, written, graphic and visual form, including electronic data and oral presentations. Plagiarism occurs when the origin of the material used is not appropriately cited.

Student signature/s

I/we declare that I/we have read and understood the declaration and statement of authorship.

(1)	Luong Trac Duc Anh	(4)	
(2)		(5)	
(3)		(6)	

Further information relating to the penalties for plagiarism, which range from a formal caution to expulsion from the University is contained on the Current Students website at **www.swin.edu.au/student/**

Copies of this form can be downloaded from the Student Forms web page at **www.swinburne.edu.au/studentforms/**

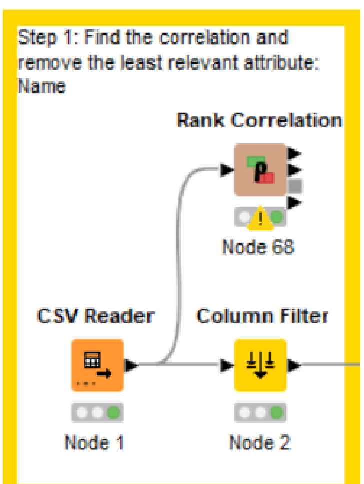
This is a report to demonstrate each step in Assignment 2. The assignment is about data cleaning and implementation of the Naïve Bayes classifier and Random Forest classifier with the cleaned data.

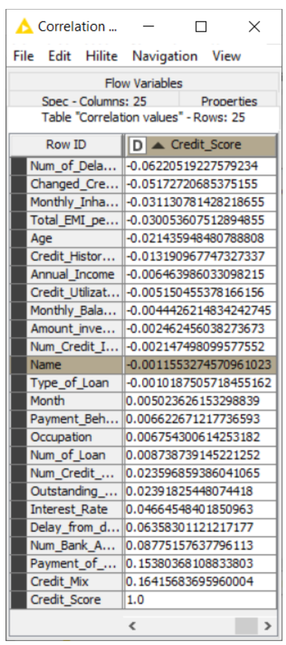
Data Cleaning

Data cleaning is the process of removing and/or fixing errors, missing values, and converting attributes to the suitable data types in a dataset. This process will make the data usable for data analysis that can produce accurate and consistent results. If the data is raw and dirty, errors and misleading values can happen during the process of prediction and conclusion. The process also relates to privacy protocols, which will be discussed further with an example in step 1.

1. The given dataset contains many invalid values and needs to be refactored or removed. The goal of predicting the credit score can also be affected by attributes that are irrelevant and have no correlation with the credit score. The one attribute that we need to remove is Name, which is a column that contains categorical data.

Using the Rank Correlation node, we can figure out which attribute has the least correlation with Credit Score (The closer it is to 0, the less relevant the 2 attributes are, 1 is a perfect correlation and -1 is a perfect negative correlation). We can see that Name and Type of Loan are the attributes that is the closest to 0, however, Type of Loan has not been cleaned yet and we will need this attribute is the later stages of the assignment. Therefore, only Name should be removed.




Correlation ...
File Edit Hilite Navigation View
Flow Variables
Spec - Columns: 25 | Properties
Table "Correlation values" - Rows: 25

Row ID	D Credit_Score
Num_of_Dela...	-0.06220519227579234
Changed_Cre...	-0.05172720685375155
Monthly_Inha...	-0.031130781428218655
Total_EMI_pe...	-0.030053607512894855
Age	-0.021435948480788808
Credit_Histor...	-0.013190967747327337
Annual_Income	-0.006463986033098215
Credit_Utilizat...	-0.005150455378166156
Monthly_Bala...	-0.0044426214834242745
Amount_inve...	-0.002462456038273673
Num_Credit_I...	-0.002147498099577552
Name	-0.0011553274570961023
Type_of_Loan	-0.0010187505718455162
Month	0.005023626153298839
Payment_Beh...	0.006622671217736593
Occupation	0.006754300614253182
Num_of_Loan	0.008738739145221252
Num_Credit_...	0.023596859386041065
Outstanding_...	0.02391825448074418
Interest_Rate	0.04664548401850963
Delay_from_d...	0.06358301121217177
Num_Bank_A...	0.08775157637796113
Payment_of_...	0.15380368108833803
Credit_Mix	0.16415683695960004
Credit_Score	1.0

By eliminating the name of the customer, the analysis will shift its focus on the patterns and trends that exist within the data, rather than individual customers. The data can now be easily analyzed with improved accuracy, reliability, and is not bias to any group of names. There are also privacy and security reasons related to this decision, however, only data analytics will be focused on this report.

2. After removing the Name attribute, we need to remove rows that contain missing values. Many rows that contain invalid values also need to be removed. For this step, I used the Rule-based

Row Filter. After filtering the data went from 100000 rows to 93512 rows (**6488 rows removed**). The command for this node is (Exclude TRUE matches):

MISSING $Month$ => TRUE

MISSING Age => TRUE

MISSING $Occupation$ => TRUE

MISSING $Annual_Income$ => TRUE

$Monthly_Inhand_Salary$ < 0 => TRUE

MISSING $Num_Bank_Accounts$ OR $Num_Bank_Accounts$ < 0 => TRUE

MISSING Num_Credit_Card OR Num_Credit_Card < 0 => TRUE

MISSING $Interest_Rate$ => TRUE

MISSING Num_of_Loan => TRUE

MISSING $Delay_from_due_date$ => TRUE

MISSING $Changed_Credit_Limit$ OR $Changed_Credit_Limit$ LIKE "_" => TRUE

MISSING $Credit_Mix$ => TRUE

MISSING $Outstanding_Debt$ => TRUE

MISSING $Credit_Utilization_Ratio$ => TRUE

MISSING $Credit_History_Age$ => TRUE

MISSING $Payment_of_Min_Amount$ => TRUE

MISSING $Total_EMI_per_month$ => TRUE

MISSING $Amount_invested_monthly$ => TRUE

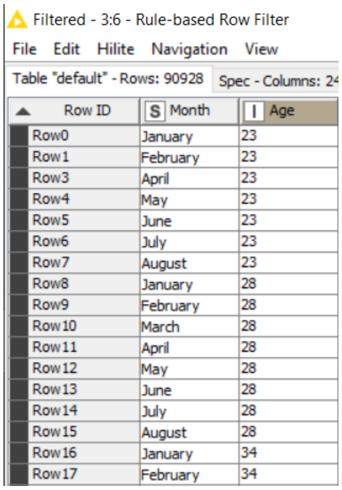
MISSING $Payment_Behaviour$ => TRUE

Figure 1: Number of rows and columns

3. After filtering out the missing values, we will go more detail into each attribute. Foe the Age attribute, we need to eliminate symbols that are not related to numerical values and convert the column into number format. Logically, rows with Age values that is smaller than 0 and larger than 120 will be removed. The needed nodes and commands regarding the Age column is currently containing string values are:

Sequence	Node	Command
1	String Manipulation	Expression: regexReplace(Age, "[^0-9.]", "") Replace Column: Age
2	String to Number	Type: Number (integer)

		Include: Age
3	Rule-based Row Filter	Age < 0 OR Age > 120 => TRUE (Exclude True matches)

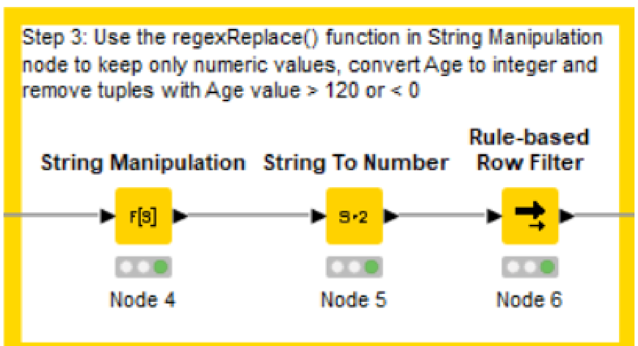


Filtered - 3:6 - Rule-based Row Filter

File Edit Hilite Navigation View

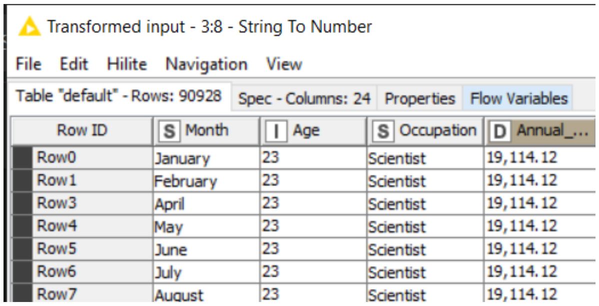
Table "default" - Rows: 90928 Spec - Columns: 24

Row ID	S Month	I Age
Row0	January	23
Row1	February	23
Row3	April	23
Row4	May	23
Row5	June	23
Row6	July	23
Row7	August	23
Row8	January	28
Row9	February	28
Row10	March	28
Row11	April	28
Row12	May	28
Row13	June	28
Row14	July	28
Row15	August	28
Row16	January	34
Row17	February	34



4. Similarly, we need to remove symbols that are not numeric for the Annual Income column. The logic is similar to Age, where we will use a regex replace function to filter out the symbols and keep only numbers from 0 to 9 and the decimal point. After that, the column is converted into the double number format. The needed nodes and commands are:

Sequence	Node	Command
1	String Manipulation	Expression: regexReplace($Annual_Income$, "[^0-9.]", "") Replace Column: Annual_Income
2	String to Number	Type: Number (double) Include: Annual_Income

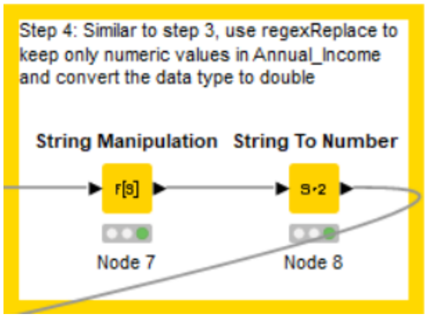


Transformed input - 3:8 - String To Number

File Edit Hilite Navigation View

Table "default" - Rows: 90928 | Spec - Columns: 24 | Properties | Flow Variables

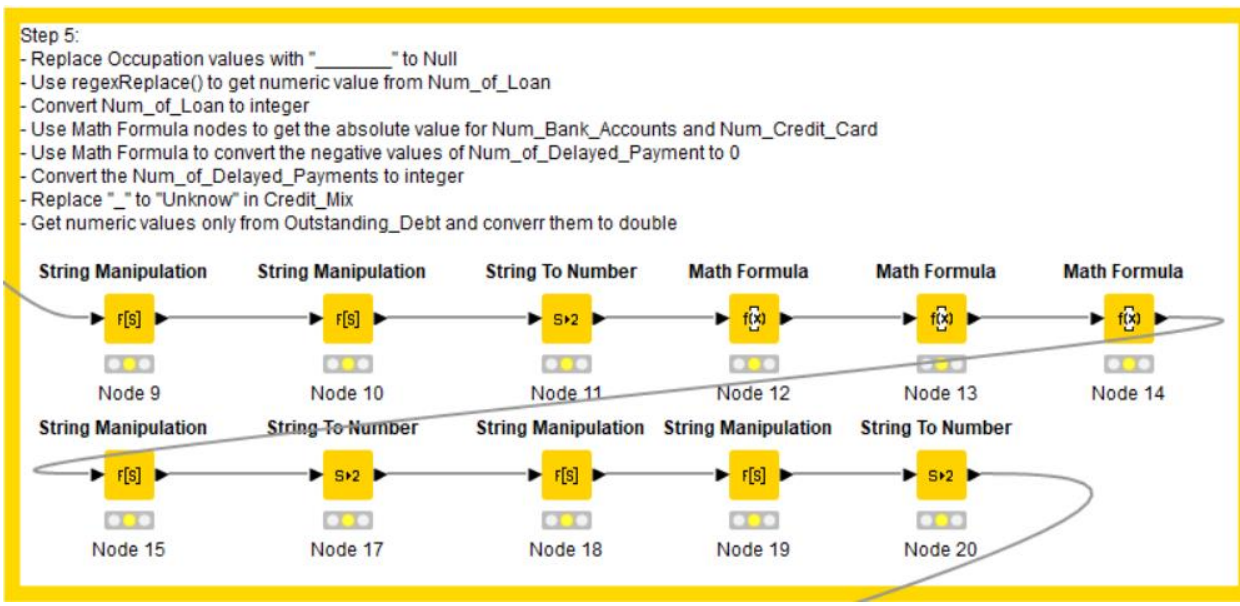
Row ID	S Month	I Age	S Occupation	D Annual_...
Row0	January	23	Scientist	19,114.12
Row1	February	23	Scientist	19,114.12
Row3	April	23	Scientist	19,114.12
Row4	May	23	Scientist	19,114.12
Row5	June	23	Scientist	19,114.12
Row6	July	23	Scientist	19,114.12
Row7	August	23	Scientist	19,114.12

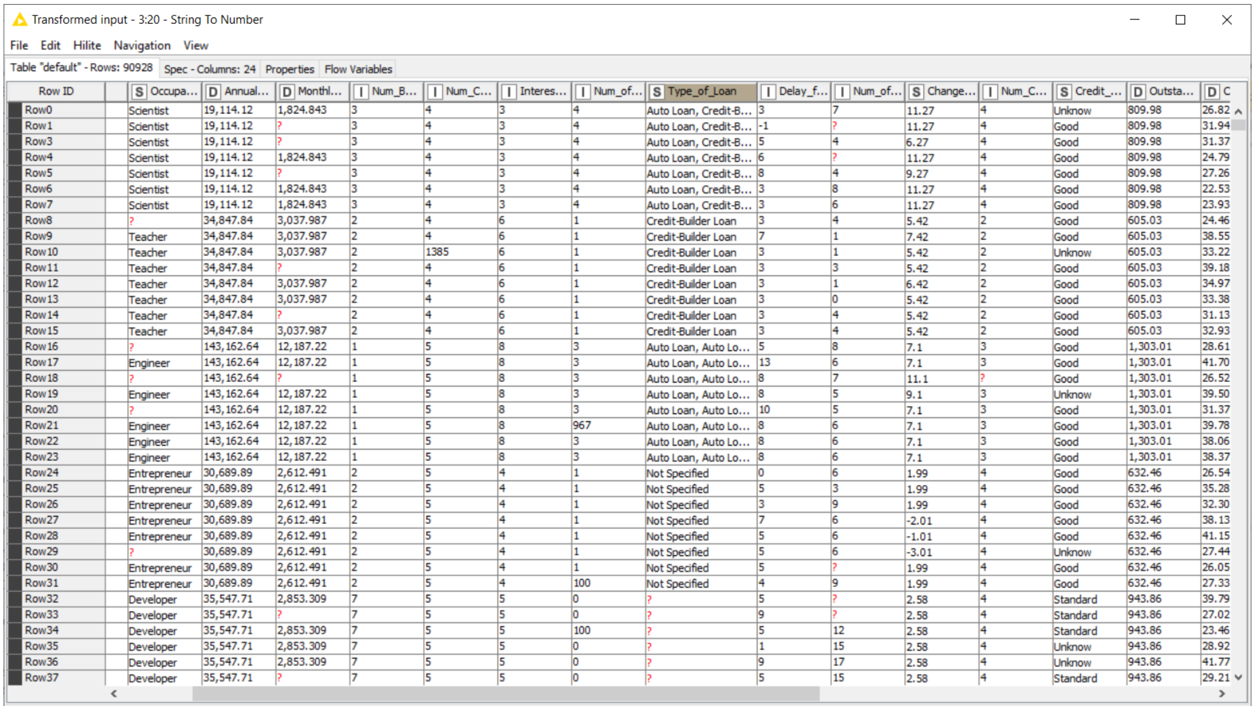


5. For the Occupation attribute, the cells with underscore symbols "_" need to be converted to Null values. Non-numerical symbols in the Number of Loan column need to be removed and the column can then be converted to integer value. The Number of Bank Accounts and Number of Credit Card will take the absolute value, which means negative values will be converted to positive and positive values remain the same. The Number of Loan column has negative values, which will then be converted to 0. Cleaning redundant symbols and converting to integer is also required for Number of Delayed payment. We do the same for Outstanding Debt and convert it into double number format. Underscore value "_" in Outstanding Debt will be changed to Unknow. The following nodes and commands are used:

Sequence	Node	Command
1	String Manipulation	Expression: toNull(replace($Occupation$, "_______", "")) Replace Column: Occupation
2	String Manipulation	Expression: regexReplace(Num_of_Loan, "[^0-9.]", "") Replace Column : Num_of_Loan
3	String to Number	Type: Number (integer) Include: Num_of_Loan
4	Math Formula	Expression: abs($Num_Bank_Accounts$) Replace Column: Num_Bank_Accounts Convert to Int
5	Math Formula	Expression: abs (Num_Credit_Card) Replace Column: Num_Credit_Card Convert to Int
6	Math Formula	Expression: max_in_args(0, Num_of_Loan) Replace Column: Num_of_Loan Convert to Int

7	String Manipulation	Expression: regexReplace($Num_of_Delayed_Payment$, "[^0-9.]", "") Replace Column: Num_of_Delayed_Payment
8	String to Number	Type: Number (integer) Include: Num_of_Delayed_Payment
9	String Manipulation	Expression: replace($Credit_Mix$, "_", "Unknow") Replace Column: Credit_Mix
10	String Manipulation	Expression: regexReplace($Outstanding_Debt$, "[^0-9.]", "") Replace Column: Outstanding_Debt
11	String to Number	Type: Number (double) Replace Column: Outstanding_Debt




Transformed input - 3:20 - String To Number

File Edit Hilite Navigation View

Table "default" - Rows: 90928 | Spec - Columns: 24 | Properties | Flow Variables

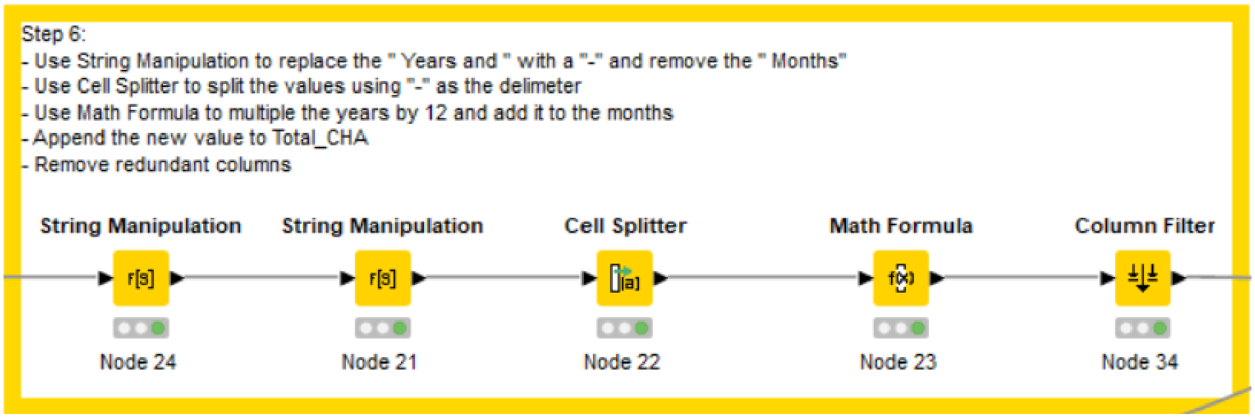
Row ID	S Occupa...	D Annual...	D Monthl...	I Num_B...	I Num_C...	I Interes...	I Num_of...	S Type_of_Loan	I Delay_f...	I Num_of...	S Change...	I Num_C...	S Credit_...	D Outsta...	D C
Row0	Scientist	19,114.12	1,824.843	3	4	3	4	Auto Loan, Credit-B...	3	7	11.27	4	Unknow	809.98	26.82
Row1	Scientist	19,114.12	?	3	4	3	4	Auto Loan, Credit-B...	-1	?	11.27	4	Good	809.98	31.94
Row3	Scientist	19,114.12	?	3	4	3	4	Auto Loan, Credit-B...	5	4	6.27	4	Good	809.98	31.37
Row4	Scientist	19,114.12	1,824.843	3	4	3	4	Auto Loan, Credit-B...	6	?	11.27	4	Good	809.98	24.79
Row5	Scientist	19,114.12	?	3	4	3	4	Auto Loan, Credit-B...	8	4	9.27	4	Good	809.98	27.26
Row6	Scientist	19,114.12	1,824.843	3	4	3	4	Auto Loan, Credit-B...	3	8	11.27	4	Good	809.98	22.53
Row7	Scientist	19,114.12	1,824.843	3	4	3	4	Auto Loan, Credit-B...	3	6	11.27	4	Good	809.98	23.93
Row8	?	34,847.84	3,037.987	2	4	6	1	Credit-Builder Loan	3	4	5.42	2	Good	605.03	24.46
Row9	Teacher	34,847.84	3,037.987	2	4	6	1	Credit-Builder Loan	7	1	7.42	2	Good	605.03	38.55
Row10	Teacher	34,847.84	3,037.987	2	1385	6	1	Credit-Builder Loan	3	1	5.42	2	Unknow	605.03	33.22
Row11	Teacher	34,847.84	?	2	4	6	1	Credit-Builder Loan	3	3	5.42	2	Good	605.03	39.18
Row12	Teacher	34,847.84	3,037.987	2	4	6	1	Credit-Builder Loan	3	1	6.42	2	Good	605.03	34.97
Row13	Teacher	34,847.84	3,037.987	2	4	6	1	Credit-Builder Loan	3	0	5.42	2	Good	605.03	33.38
Row14	Teacher	34,847.84	?	2	4	6	1	Credit-Builder Loan	3	4	5.42	2	Good	605.03	31.13
Row15	Teacher	34,847.84	3,037.987	2	4	6	1	Credit-Builder Loan	3	4	5.42	2	Good	605.03	32.93
Row16	?	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	5	8	7.1	3	Good	1,303.01	28.61
Row17	Engineer	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	13	6	7.1	3	Good	1,303.01	41.70
Row18	?	143,162.64	?	1	5	8	3	Auto Loan, Auto Lo...	8	7	11.1	?	Good	1,303.01	26.52
Row19	Engineer	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	8	5	9.1	3	Unknow	1,303.01	39.50
Row20	?	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	10	5	7.1	3	Good	1,303.01	31.37
Row21	Engineer	143,162.64	12,187.22	1	5	8	967	Auto Loan, Auto Lo...	8	6	7.1	3	Good	1,303.01	39.78
Row22	Engineer	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	8	6	7.1	3	Good	1,303.01	38.06
Row23	Engineer	143,162.64	12,187.22	1	5	8	3	Auto Loan, Auto Lo...	8	6	7.1	3	Good	1,303.01	38.37
Row24	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	0	6	1.99	4	Good	632.46	26.54
Row25	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	5	3	1.99	4	Good	632.46	35.28
Row26	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	3	9	1.99	4	Good	632.46	32.30
Row27	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	7	6	-2.01	4	Good	632.46	38.13
Row28	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	5	6	-1.01	4	Good	632.46	41.15
Row29	?	30,689.89	2,612.491	2	5	4	1	Not Specified	5	6	-3.01	4	Unknow	632.46	27.44
Row30	Entrepreneur	30,689.89	2,612.491	2	5	4	1	Not Specified	5	?	1.99	4	Good	632.46	26.05
Row31	Entrepreneur	30,689.89	2,612.491	2	5	4	100	Not Specified	4	9	1.99	4	Good	632.46	27.33
Row32	Developer	35,547.71	2,853.309	7	5	5	0	?	5	?	2.58	4	Standard	943.86	39.79
Row33	Developer	35,547.71	?	7	5	5	0	?	9	?	2.58	4	Standard	943.86	27.02
Row34	Developer	35,547.71	2,853.309	7	5	5	100	?	5	12	2.58	4	Standard	943.86	23.46
Row35	Developer	35,547.71	2,853.309	7	5	5	0	?	1	15	2.58	4	Unknow	943.86	28.92
Row36	Developer	35,547.71	2,853.309	7	5	5	0	?	9	17	2.58	4	Unknow	943.86	41.77
Row37	Developer	35,547.71	?	7	5	5	0	?	5	15	2.58	4	Standard	943.86	29.21

Figure 2: Step 5

6. For step 6, we need to change the column Credit History Age to only counting months (years will be multiple with 12 and add to the months). To do this, we use a String Manipulation node to replace “NA” with 0. After that, we replace the “ Years and ” with a hyphen “-” and remove the “ Months” string. For example, “12 Years and 8 Months” will now be 12-8, append to a new column named Total_CHA. After that, we use a Cell Splitter to split the Total_CHA column using the delimiter “-“. We also need to remove any white space left in the column cells. The good thing is the elements in our new array are automatically converted into the correct data type, in this case, integer. We will get a new array of Total CHA in each row with the years at the 0 index and the months at the 1 index. After that, we use a Math Formula node to multiply the index 0 columns by 12 and add it to the element at index 1 and append it back to Total_CHA. Finally, we remove the 2 redundant array elements using a Column Filter. The nodes and commands used are:

Sequence	Node	Command
1	String Manipulation	Expression: replace($Credit_History_Age$, "NA", "0") Replace Column: Credit_History_Age
2	String Manipulation	Expression: replace(replace($Credit_History_Age$, " Years and ", "-"), " Months", "") Append Column: Total_CHA
3	Cell Splitter	Select a column: Total_CHA Enter a delimeter: - Remove leading and trailing white space chars (trim)
4	Math Formula	Expression:

		$Total_CHA_Arr[0]$*12+$Total_CHA_Arr[1]$ Replace Column: Total_CHA Convert to Int
5	Column Filter	Exclude: Total_CHA_Arr[0], Total_CHA_Arr[1] Note: To remove the redundant array column created by the Cell Splitter node

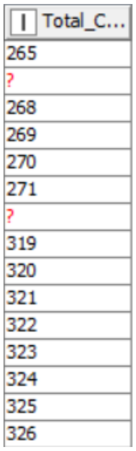


S Total_CHA	I Total_CHA_Arr[0]	I Total_CHA_Arr[1]
22-5	22	5
22-6	22	6
22-7	22	7
0	0	?
26-7	26	7
26-8	26	8
26-9	26	9
26-10	26	10
26-11	26	11
27-0	27	0
27-1	27	1
27-2	27	2
17-9	17	9
17-10	17	10
17-11	17	11

Figure 3: Column Splitter

I Total_CHA	I Total_CHA_Arr[0]	I Total_CHA_Arr[1]
265	22	1
?	0	?
268	22	4
269	22	5
270	22	6
271	22	7
?	0	?
319	26	7
320	26	8
321	26	9
322	26	10
323	26	11
324	27	0
325	27	1
326	27	2
213	17	9
214	17	10
215	17	11

Figure 4: Math Formula

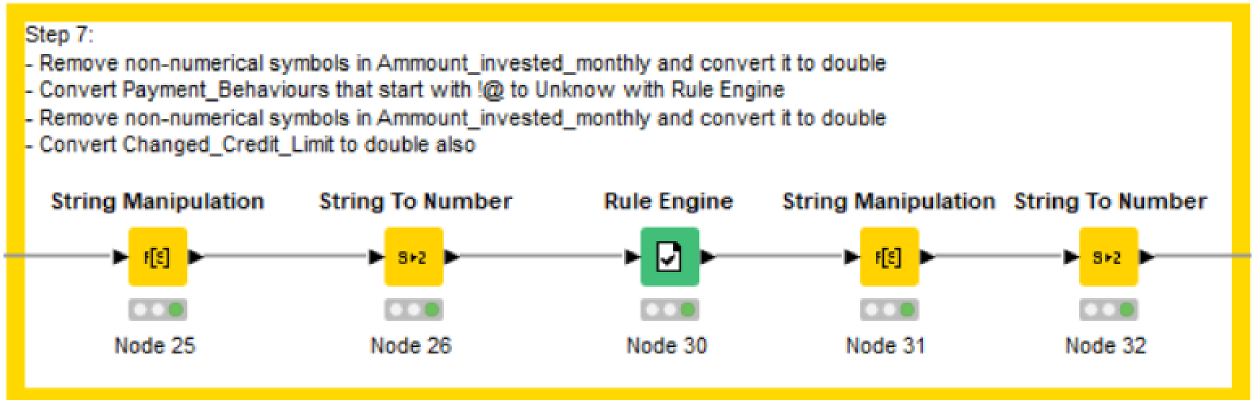


Total_C...
265
?
268
269
270
271
?
319
320
321
322
323
324
325
326

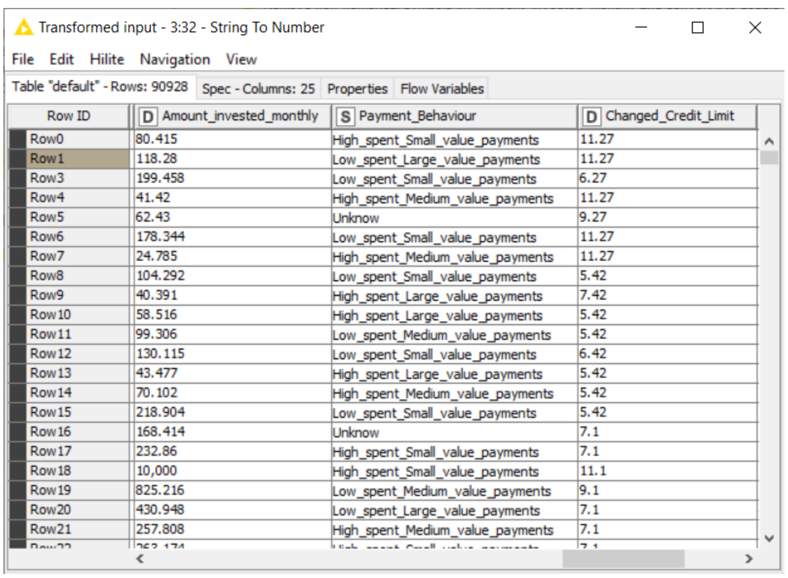
Figure 5: Column Filter

7. For step 7, we need to remove the symbols that are not numerical in the Amount invested monthly and convert it to the double data type. The Payment Behaviour cells that contains !@ at the start will be filtered and convert to the double data type. The Changed Credit Limit is will also go through this process.

Sequence	Node	Command
1	String Manipulation	Expression: regexReplace($Amount_invested_monthly$, "[^0-9.]", "") Replace Column: Amount_invested_monthly
2	String To Number	Type: Number (double) Include: Amount_invested_monthly
3	Rule Engine	Expression: $Payment_Behaviour$ LIKE "*!@*" => "Unknow" TRUE => $Payment_Behaviour$ Replace Column: Payment_Behaviour
4	String Manipulation	Expression: regexReplace($Monthly_Balance$, "[^0-9.]", "") Replace Column: Monthly_Balance
5	String to Number	Type: Number (double) Include: Monthly_Balance, Changed_Credit_Limit



Output:



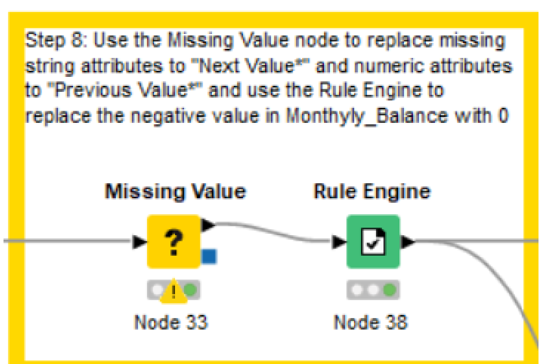
Transformed input - 3:32 - String To Number

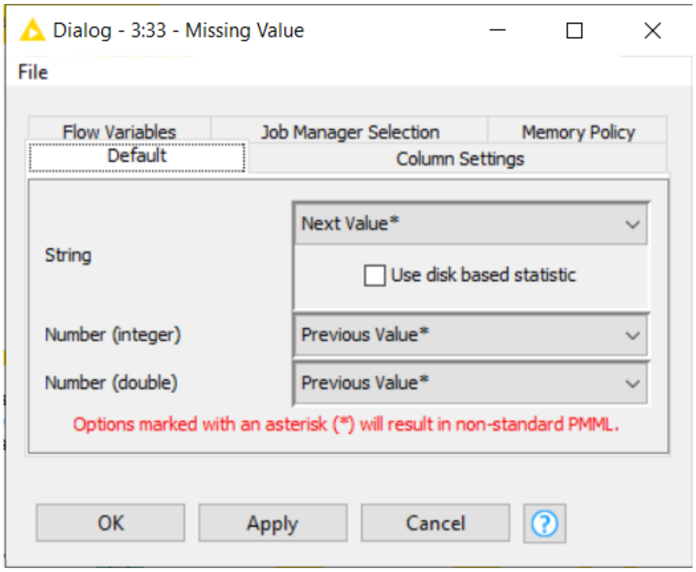
File Edit Hilite Navigation View

Table "default" - Rows: 90928 | Spec - Columns: 25 | Properties | Flow Variables

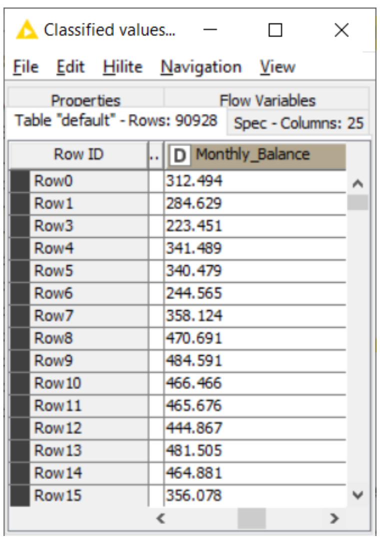
Row ID	D Amount_invested_monthly	S Payment_Behaviour	D Changed_Credit_Limit
Row0	80.415	High_spent_Small_value_payments	11.27
Row1	118.28	Low_spent_Large_value_payments	11.27
Row3	199.458	Low_spent_Small_value_payments	6.27
Row4	41.42	High_spent_Medium_value_payments	11.27
Row5	62.43	Unknow	9.27
Row6	178.344	Low_spent_Small_value_payments	11.27
Row7	24.785	High_spent_Medium_value_payments	11.27
Row8	104.292	Low_spent_Small_value_payments	5.42
Row9	40.391	High_spent_Large_value_payments	7.42
Row10	58.516	High_spent_Large_value_payments	5.42
Row11	99.306	Low_spent_Medium_value_payments	5.42
Row12	130.115	Low_spent_Small_value_payments	6.42
Row13	43.477	High_spent_Large_value_payments	5.42
Row14	70.102	High_spent_Medium_value_payments	5.42
Row15	218.904	Low_spent_Small_value_payments	5.42
Row16	168.414	Unknow	7.1
Row17	232.86	High_spent_Small_value_payments	7.1
Row18	10,000	High_spent_Small_value_payments	11.1
Row19	825.216	Low_spent_Medium_value_payments	9.1
Row20	430.948	Low_spent_Large_value_payments	7.1
Row21	257.808	High_spent_Medium_value_payments	7.1

8. The nodes and commands for step 8:





Output for the replacement of negative Monthly Balance value to 0:



9. In step 9, we need to convert the Type of Loan attribute so that if the original value has more than 1 type and those types are separated by comma, we will only keep the 1st part, remove everything that is behind the comma. To do this, we have 2 methods:

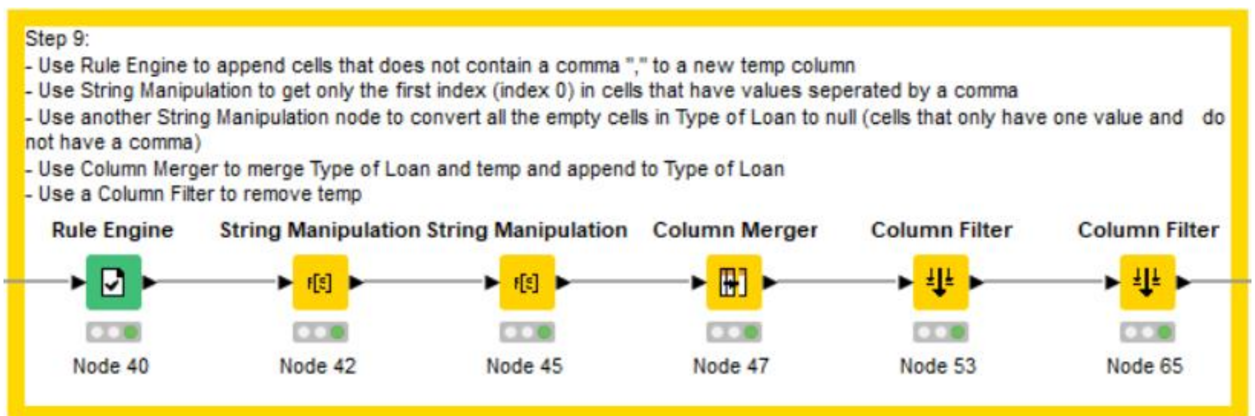
1st method:

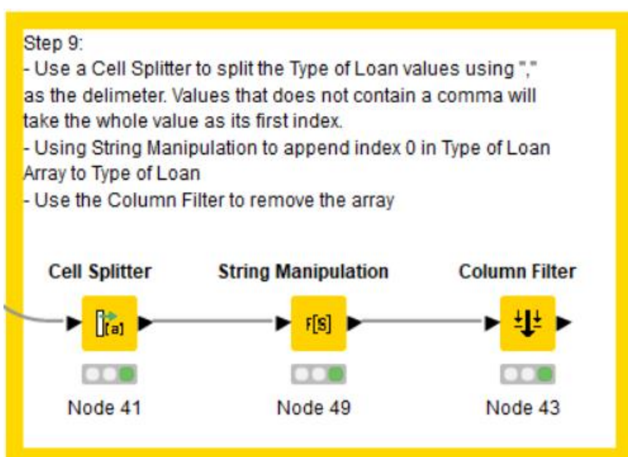
Sequence	Node	Command
1	Rule Enginge	Expression: NOT $Type_of_Loan$ LIKE "*,*" => $Type_of_Loan$ Append column: temp
2	String Manipulation	Expression: substr($Type_of_Loan$, 0, indexOf($Type_of_Loan$, ",")) Replace Column: Type_of_Loan
3	String Manipulation	toNull($Type_of_Loan$)

4	Column Merger	Primary Column: Type_of_Loan Secondary Column: temp Output replacement: Replace primary column
5	Column Filter	Exclude: temp

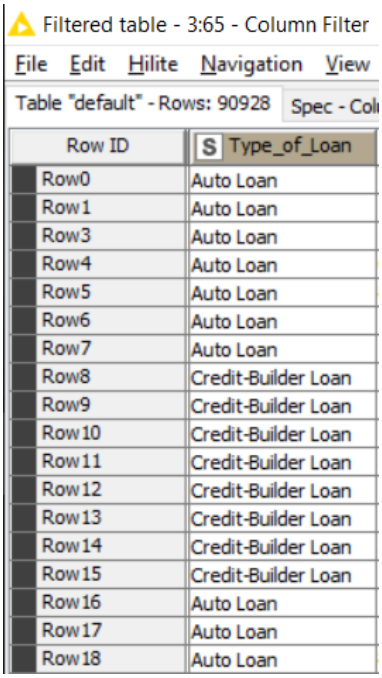
2[nd] method:

Sequence	Node	Command
1	Cell Splitter	Select a column: Type_of_Loan Enter a delimeter: , Remove leading and trailing white space chars (trim)
2	String Manipulation	Expression: string($Type_of_Loan_Arr[0]$) Replace column: Type_of_Loan
3	Column Filter	Exclude: Type_of_Loan_Arr[0], Type_of_Loan_Arr[1], Type_of_Loan_Arr[2], Type_of_Loan_Arr[3], Type_of_Loan_Arr[4], Type_of_Loan_Arr[5], Type_of_Loan_Arr[6], Type_of_Loan_Arr[7], Type_of_Loan_Arr[8]

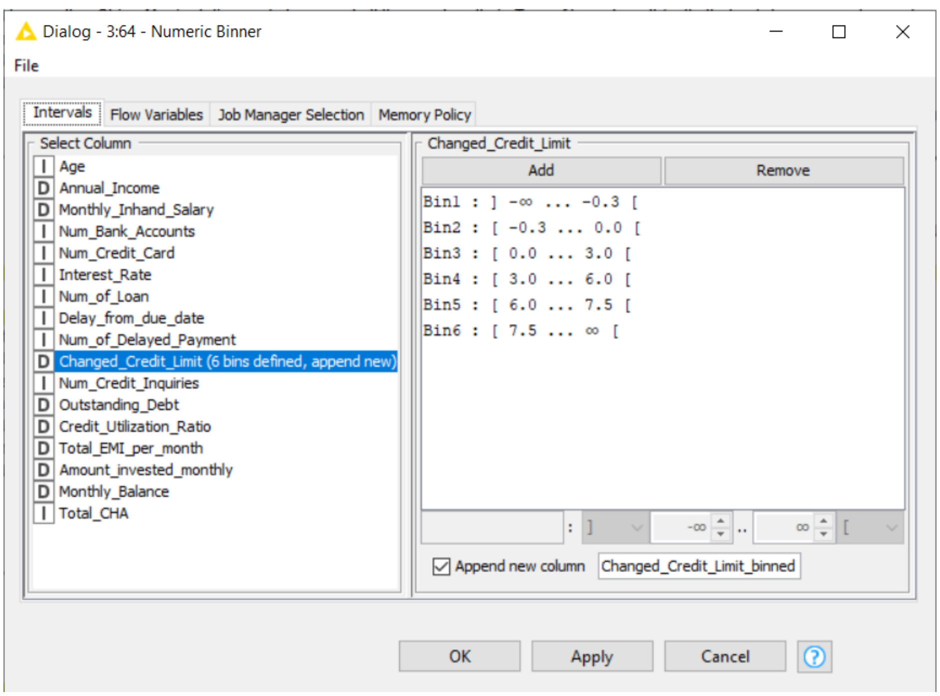


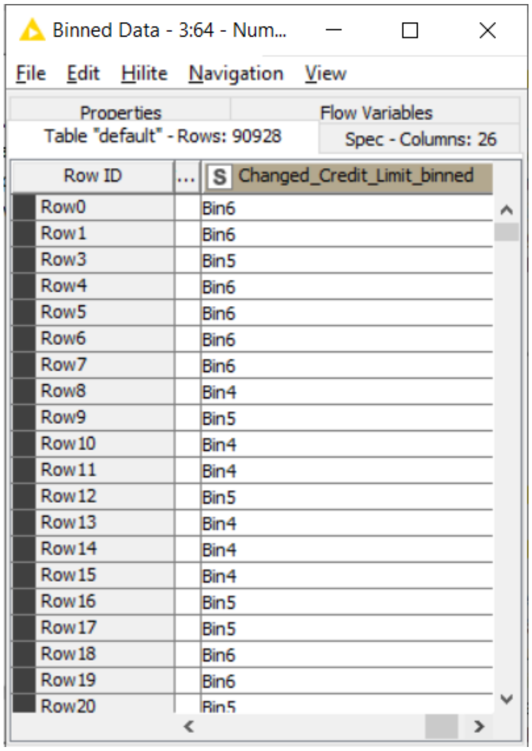


Method 2 is generally less complex but they both produce the same output:

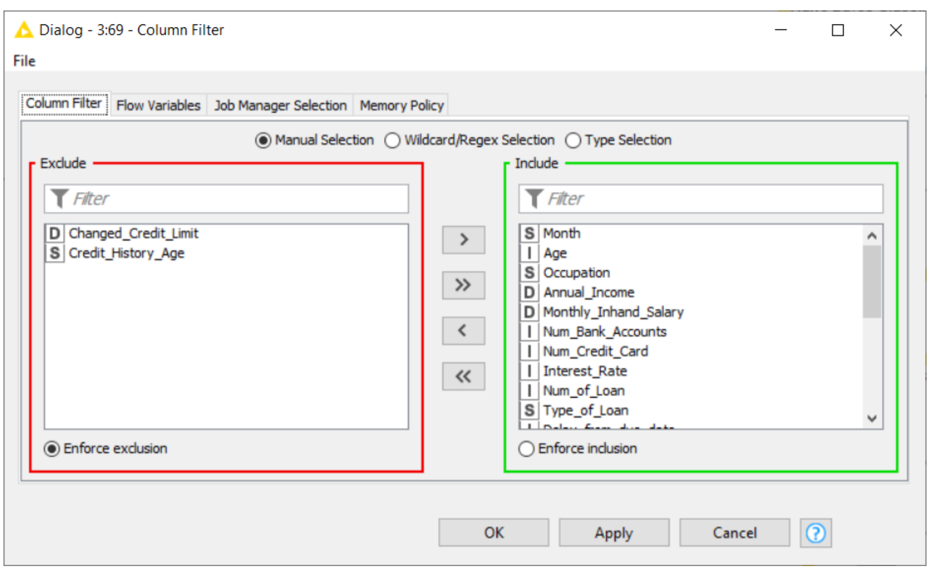


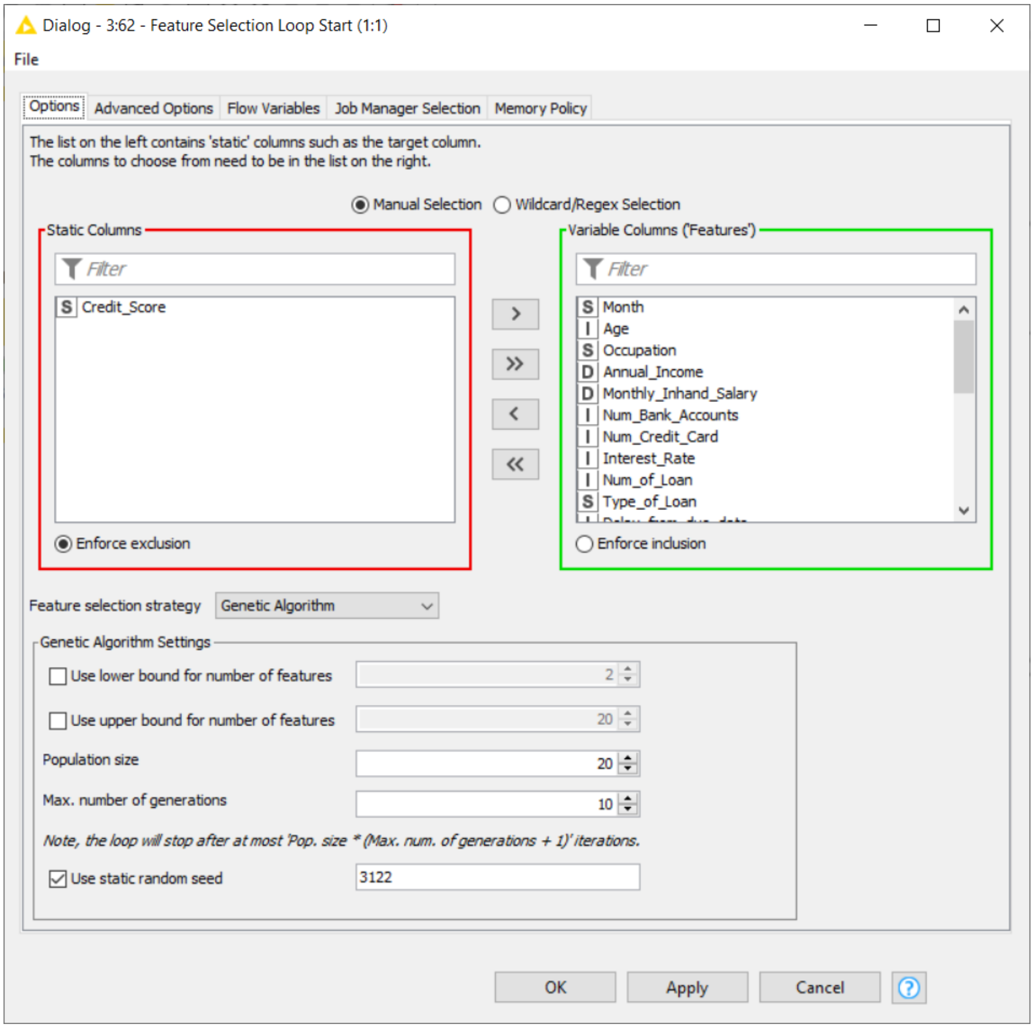
10. For step 10, we need to bin the Changed Credit Limit into 6 ranges that are defined in the requirement. Here is the screenshot for the Numeric Binner node:



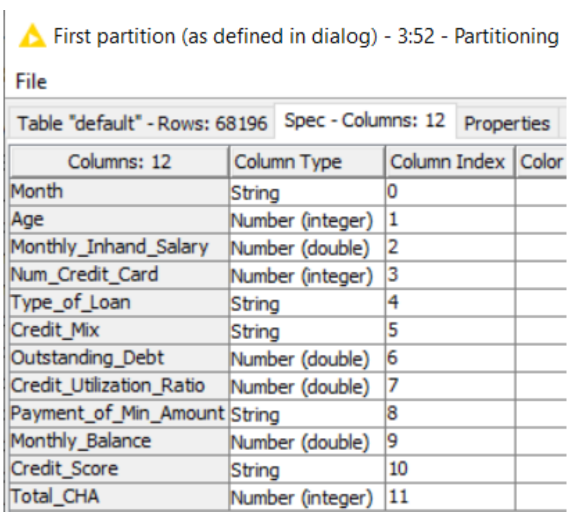


11. For the last step in Data Cleaning, we need to remove the attributes that are not necessary for our Credit Score prediction. First, we need to remove temporarily created and useless columns using a Column Filter. These include Credit History Age and Changed Credit Limit as they were replaced by Total CHA and Changed Credit Limit binned respectively. The Credit Score will be used as the Static Column, meaning it will be excluded from our Variable Features. The node Feature Selection Loop Start will be used. We will use the Genetic Algorithm feature selection strategy with 3122 as the random seed. The settings for the nodes are screenshotted below:





After partitioning with linear sampling and 75% of the data in the training set, 25% of the data in the test set, the filtered table will have 11 columns excluding the Credit Score as out class label and 68196 rows.

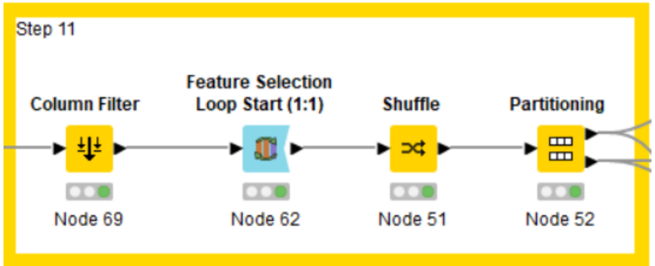


First partition (as defined in dialog) - 3:52 - Partitioning

File

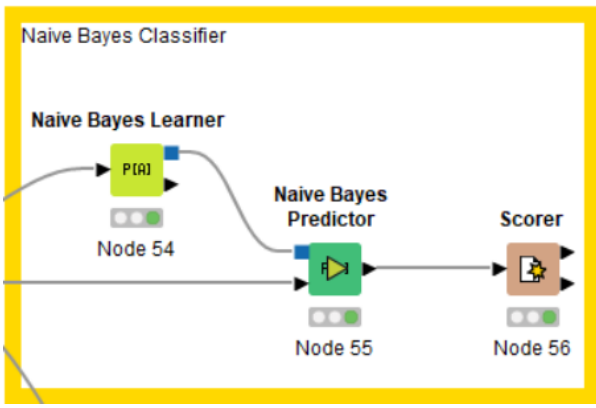
Table "default" - Rows: 68196 | Spec - Columns: 12 | Properties

Columns: 12	Column Type	Column Index	Color
Month	String	0	
Age	Number (integer)	1	
Monthly_Inhand_Salary	Number (double)	2	
Num_Credit_Card	Number (integer)	3	
Type_of_Loan	String	4	
Credit_Mix	String	5	
Outstanding_Debt	Number (double)	6	
Credit_Utilization_Ratio	Number (double)	7	
Payment_of_Min_Amount	String	8	
Monthly_Balance	Number (double)	9	
Credit_Score	String	10	
Total_CHA	Number (integer)	11	



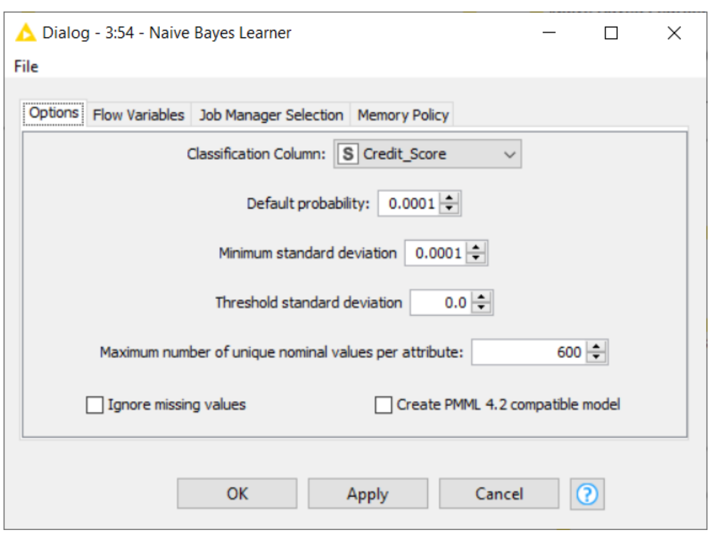
Naïve Bayes Classifier

1. The workflow for the Naïve Bayes classifier is:



After partitioning, the training dataset will be fetched to the Learner node, while the test dataset will be fetched to the Predictor node. After that, we will use a Scorer node to view the result and accuracy of our model.

2. The settings for The Learner node are:



3. The confusion matrix and accuracy statistics are respectively:

Confusion matrix - 3:56 - Scorer

File Edit Hilite Navigation View

Table "spec_name" - Rows: 3 | Spec - Columns: 3 | Properties | Flow Variables

Row ID	Good	Standard	Poor
Good	3186	817	100
Standard	2663	7252	2168
Poor	1023	2139	3384

Accuracy statistics - 3:56 - Scorer

File Edit Hilite Navigation View

Table "default" - Rows: 4 | Spec - Columns: 11 | Properties | Flow Variables

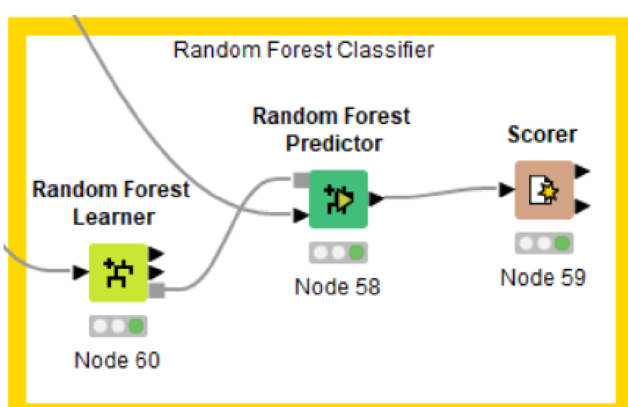
Row ID	TruePo...	FalsePo...	TrueNe...	FalseN...	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
Good	3186	3686	14943	917	0.777	0.464	0.777	0.802	0.581	?	?
Standard	7252	2956	7693	4831	0.6	0.71	0.6	0.722	0.651	?	?
Poor	3384	2268	13918	3162	0.517	0.599	0.517	0.86	0.555	?	?
Overall	?	?	?	?	?	?	?	?	?	0.608	0.383

The bank would want to lower the risk by giving loans to customers that have a good Credit Scores. From the results screenshotted above, using the Naïve Bayes classifier did not produce a satisfactory result. When it comes to giving loans to customers with good Credit Scores, the Precision metric number should be taken into consideration. Precision is calculated as TP/(TP+FP), therefore it will measure the cases that are predicted as positive and are correctly positive. The number will calculate the percentage of the bank only lending money to customers with Good Credit Score and will not label customers with Standard or Poor Credit Scores to be Good. This would lower the risk of the bank giving money to the wrong customers which can result in financial burdens and losses. For the Naïve Bayes model, we got an accuracy of 0.608 overall, however, when it came to the Good Credit Score, the Precision was only 0.464, which was relatively low and indicates that the model only get the correct Good Credit Score less than half of the time, and more than half of the time it identifies a customer with Standard or Poor Credit Score as Good, which is false at an alarming rate. Therefore, this result will be not satisfactory.

4. As I have mentioned above, the measure will look at the Precision of the model. The situations that are predicted as positive and positive will be measured since precision is computed as TP/(TP+FP). The percentage of clients with Good Credit Scores who receive loans from the bank will be calculated; customers with Standard or Poor Credit Scores won't be considered good customers.

Random Forest Classifier

1. The workflow for the Random Forest classifier is:



After partitioning, the training dataset will be fetched to the Learner node, while the test dataset will be fetched to the Predictor node. After that, we will use a Scorer node to view the result and accuracy of our model.

2. The confusion matrix and accuracy statistics are respectively:

Confusion matrix - 3:59 - Scorer

File Edit Hilite Navigation View

Table "spec_name" - Rows: 3 | Spec - Columns: 3 | Properties | Flow Variables

Row ID	I Good	I Standard	I Poor
Good	2818	1260	25
Standard	912	9774	1397
Poor	179	1504	4863

Accuracy statistics - 3:59 - Scorer

File Edit Hilite Navigation View

Table "default" - Rows: 4 | Spec - Columns: 11 | Properties | Flow Variables

Row ID	I TruePo...	I FalsePo...	I TrueNe...	I FalseN...	D Recall	D Precision	D Sensitivity	D Specificity	D F-meas...	D Accuracy	D Cohen'...
Good	2818	1091	17538	1285	0.687	0.721	0.687	0.941	0.703	?	?
Standard	9774	2764	7885	2309	0.809	0.78	0.809	0.74	0.794	?	?
Poor	4863	1422	14764	1683	0.743	0.774	0.743	0.912	0.758	?	?
Overall	?	?	?	?	?	?	?	?	?	0.768	0.611

3. In this Random Forest classifier, if we look at the same statistics as we used for the Naïve Bayes classifier, which is Precision and Accuracy, Random Forest performed better at 0.721 for Precision and 0.768 for Accuracy. Both numbers trumped their Naïve Bayes counterparts. If the bank's goal was to curb the risk of financial losses by lending money to customers with Good Credit Scores, the Random Forest presented a more suitable result. The high precision of 0.721 over 0.464 is strong proof of this notion. However, there is one case that the 1st model can be more suitable, which is when the bank wants to give money to as many customers with Good Credit Score as possible. The Recall for Naïve Bayes was 0.777 while the Recall for Random Forest was 0.678. This translates to more customers being identified as having Good Credit Scores, but also given a higher chance of False Positives as the calculation for Recall is Recall = TP/(TP+TN). However, this situation is not ideal as the bank would lose more money lending money to customers with Standard and Poor Credit Scores and is only a case that we consider theoretically.
4. The class that our random forest model performed the best on was **Standard** Credit Score, as we can look at its Precision and Recall, which were highest amongst the 3 classes at 0.78 and 0.809 respectively. In particular, the highest Recall value of 0.809 describes that the model performed well when it comes to identifying all positive cases. On the other hand, the Precision value of 0.78 describes that the model can identify positive cases and at the same time avoid FP values. In conclusion, the model is best when performing on the Standard class as it avoids FP and FN values, and the bank can have accurate and informed decisions based on the statistics generated based on this class.

- The measurements for this answer were Precision and Recall.