

Cover sheet for submission of work for assessment



UNIT DETAILS

Unit name	Data Science Principles	Class day/time	COS10022.1	Office use only	
Unit code	COS10022	Assignment no.	02	Due date	10/11/2024
Name of lecturer/teacher	Mr. Minh Anh				
Tutor/marker's name					Faculty or school date stamp

STUDENT(S)

Family Name(s)	Given Name(s)	Student ID Number(s)
(1) Nguyen	Minh Duy	104974743
(2)		
(3)		
(4)		
(5)		
(6)		

DECLARATION AND STATEMENT OF AUTHORSHIP

- I/we have not impersonated, or allowed myself/ourselves to be impersonated by any person for the purposes of this assessment.
 - 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.
 - 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.
 - I/we have not previously submitted this work for this or any other course/unit.
 - 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:
- 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)		(4)	
(2)	Nguyen Minh Duy	(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/



COS10022 Data Science Principles

Assignment 2 - Semester 1, 2024

Assessment Title: Data Cleaning and Analytics

Assessment Weighting: 30%

Due Date: Sunday, 12th May 2023 at 11.59 pm (AEDT)

Assessable Item:

- One (1) piece of a written report no more than 10 pages along with the signed Assignment Cover Sheet.
- The submitted report must be checked by Turnitin, and the similarity from **not the template part** should be less than 12%.
- A KNIME workflow in Assessment 2.1.

The submitted report should answer all questions listed in the assignment task section in sequence. You must include a digitally signed Assignment Cover Sheet with your submission.

1. Follow the instructions to clean the data and answer questions. If any of the nodes you used in the workflow **has a random seed**, set **9214** to the seed to **fix the random state**. **[65 marks in total]**

- 1) Our goal is to predict the **credit score from the given data**. There is/are one (or multiple) attribute(s) which is/are significantly irrelevant to the goal. Pick the most irrelevant attribute and give a persuasive rationale for that. The excluded attribute(s) is _____, and the reason for removing it is _____. **[2.5 marks]**

Ans:


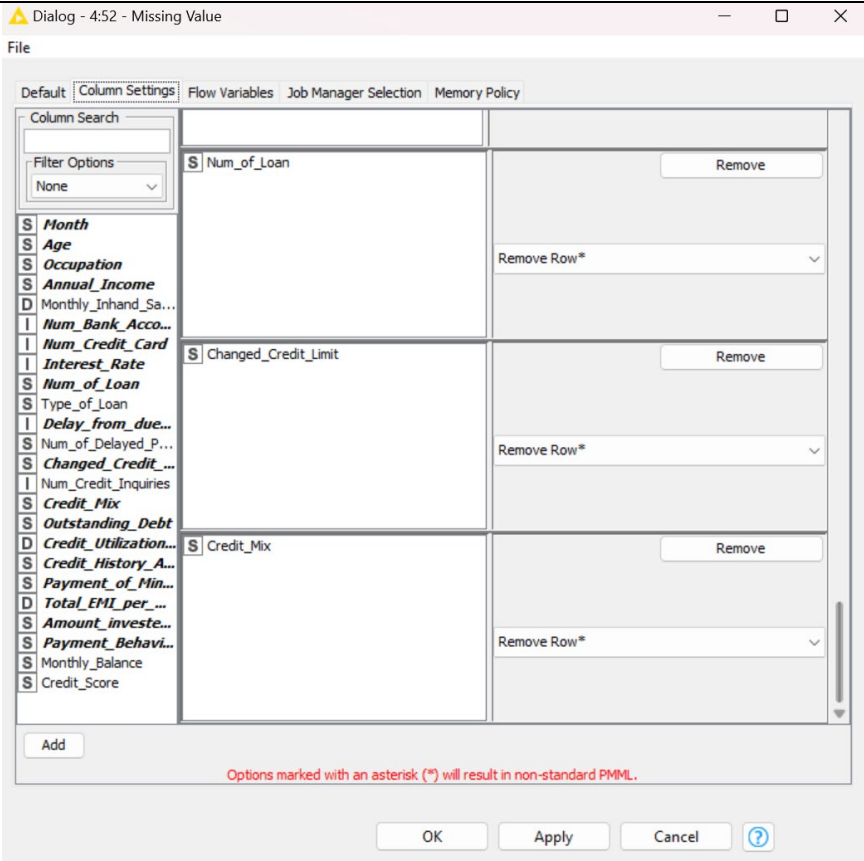
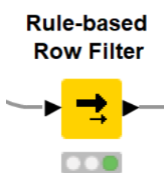
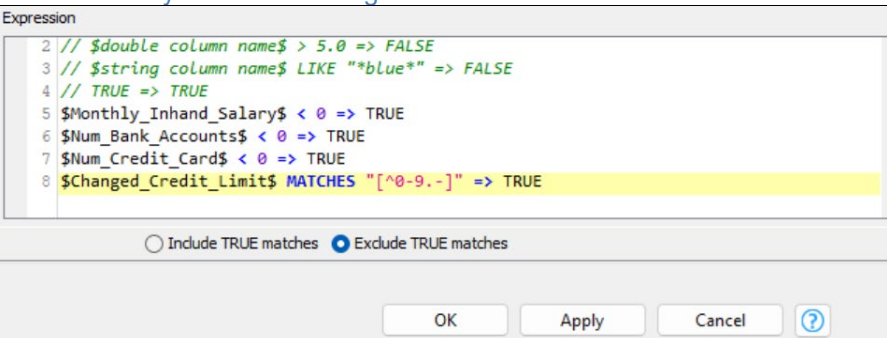
The excluded attribute is(are) **"Name"** because **The attribute "Name" is a unique identifier that doesn't affect credit score prediction, and its removal would enhance model efficiency without affecting predictive accuracy.**

- 2) After removing the selected attribute(s), let's start to remove tuples containing missing values. Remove tuples **only if any of the attributes listed below have missing values**: "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_Behaviour." Moreover, some tuples with infeasible values in the attributes, such as "Monthly_Inhand_Salary" < 0, "Num_Bank_Accounts" < 0, "Num_Credit_Card" < 0, and "Changed_Credit_Limit" contains "_", should also be removed. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[5 marks]**

Ans:


The nodes utilized are **"Missing Value"** and **"Rule-based Row Filter"**. The used commands are listed as follows:

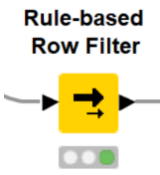
Sequence	Node	Command
----------	------	---------

1	<p>Missing Value</p> 	 <p>Set the handling method for each column to Remove Row if Missing. This will ensure any row with missing values in these columns is excluded.</p>
2	<p>Rule-based Row Filter</p> 	 <p>TRUE indicates rows that meet these conditions will be removed. \$Changed_Credit_Limit\$ MATCHES "[^0-9.-]" ensures any non-numeric or special characters (" _ ") in Changed_Credit_Limit result in row removal.</p>

- 3) Check for the "Age" attribute to eliminate symbols that are not numbers to recover the data into the usual number format. Moreover, drop the tuples whose "Age" value is lower than or equal to 0 or greater than 120. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[5 marks]**


Ans:

Sequence	Node	Command
1	<p>String Manipulation</p> 	<p>Expression</p> <pre>1 toInt(regexReplace(\$Age\$, "[^0-9.-]", ""))</pre> <p>Removes any non-numeric characters from Age and converts the result to an integer. [^0-9.-] means that all value that is NOT (^) from 0-9, or decimal point (.) or negative sign (-) are removed.</p>

2	<p>Rule-based Row Filter</p> 	<p>Expression</p> <pre> 1 // enter ordered set of rules, e.g.: 2 // \$double column name\$ > 5.0 => FALSE 3 // \$string column name\$ LIKE "*blue*" => FALSE 4 // TRUE => TRUE 5 \$Age\$ > 120 => TRUE 6 \$Age\$ <= 0 => TRUE </pre> <p>Removes rows with Age values outside the range 1-120.</p>
---	--	---



- 4) Remove the non-numerical symbol in the “Annual_Income” column and convert it to the double format. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[5 marks]**

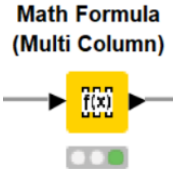
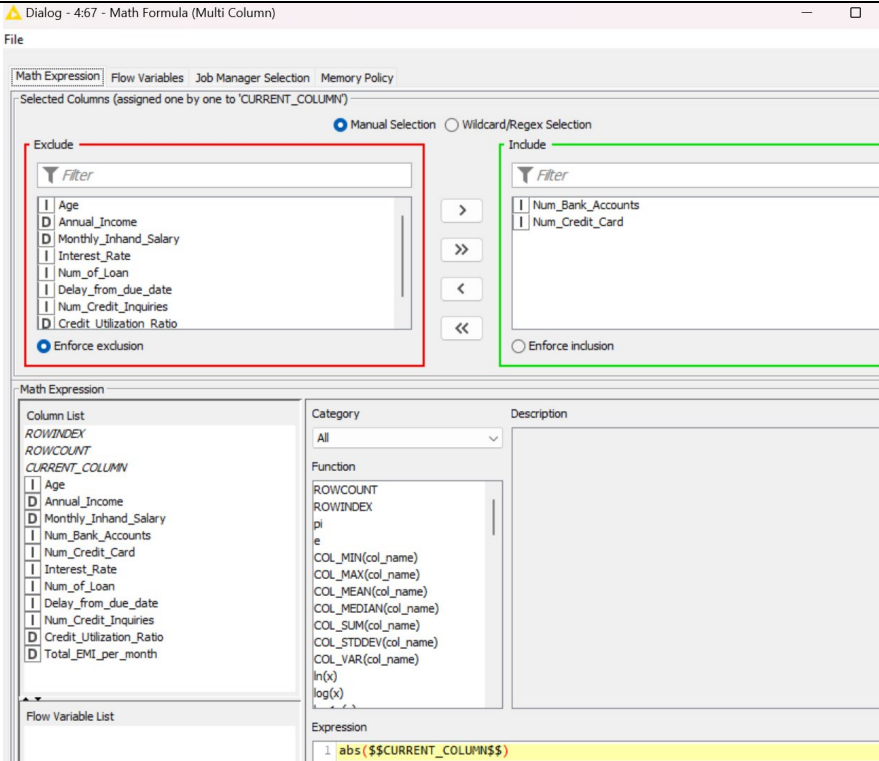
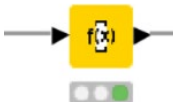
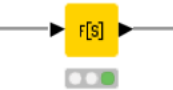
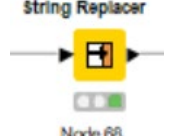
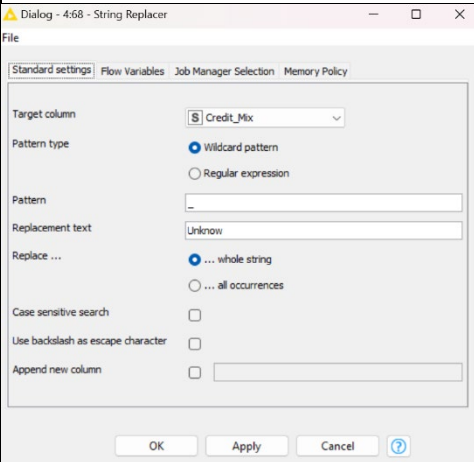
Ans:

Sequence	Node	Command
1	<p>String Manipulation</p> 	<p>Expression</p> <pre> 1 toDouble(regexReplace(\$Annual_Income\$,"[^0-9.-]", "")) </pre> <p>Removes non-numerical characters and convert to double.</p>

- 5) Convert the “_____” in the “Occupation” attribute to Null. Please note that Null is different from an empty string. Remove the non-numerical symbol in “Num_of_Loan” and convert it to integer data type. Take absolute values of attributes “Num_Bank_Accounts” and “Num_Credit_Card.” Set values to 0 for the “Num_of_Loan” attribute if the original values are negative. Remove the non-numerical symbol in “Num_of_Delayed_payment” and convert it into integer format. Set the “Credit_Mix” value to “Unknow” if the original value is “_”. Remove the non-numerical symbol in “Outstanding_Debt” and convert it into the double format. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[10 marks]**

Ans:

Sequence	Node	Command
1	<p>String Manipulation</p> 	<p>Expression</p> <pre> 1 toNull(replace(\$Occupation\$,"_____", "")) </pre> <p>Replaces any instances of “_____” in the Occupation column with an empty string (“”) and then converts the result to a null value if the field is empty.</p>
2	<p>String Manipulation</p> 	<p>Expression</p> <pre> 1 toInt(regexReplace(\$Num_of_Loan\$,"[^0-9.-]", "")) </pre> <p>Removes any non-numeric characters from Num_of_Loan and converts the cleaned result into an integer.</p>

3	<p>Math Formula (Multi-Column)</p> 	 <p>Include "Num_Bank_Accounts" and "Num_Credit_Card" and convert them to absolute values.</p>
4	<p>Math Formula</p> 	<p>Expression</p> <pre>1 if(\$Num_of_Loan\$ < 0, 0, \$Num_of_Loan\$)</pre> <p>Checks if Num_of_Loan is less than 0. If it is, the expression sets it to 0; otherwise, it keeps the original Num_of_Loan value.</p>
5	<p>String Manipulation</p> 	<p>Expression</p> <pre>1 toInt(regexReplace(\$Num_of_Delayed_Payment\$, "[^0-9.-]", ""))</pre> <p>Removes any non-numeric characters from Num_of_Delayed_Payment and then converts the cleaned result to an integer.</p>
6	<p>String Replacer</p> 	 <p>Replace any instance of "_" to "Unknown"</p>
7	<p>String Manipulation</p>	<p>Expression</p> <pre>1 toDouble(regexReplace(\$Outstanding_Debt\$, "[^0-9.-]", ""))</pre> <p>1 Removes non-numerical values and converts to doubles.</p>

- 6) Convert the “Credit_History_Age” to the count of months and store it in the integer format. For example, if the original value from a tuple is “22 Years and 1 Months”, the value will be 265 after the conversion ($22 * 12 + 1 = 265$). Store the converted result in a new attribute called “Total_CHA.” List the node(s) (in sequence) and the corresponding command(s) used in this process. **[10 marks]**

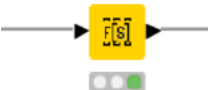
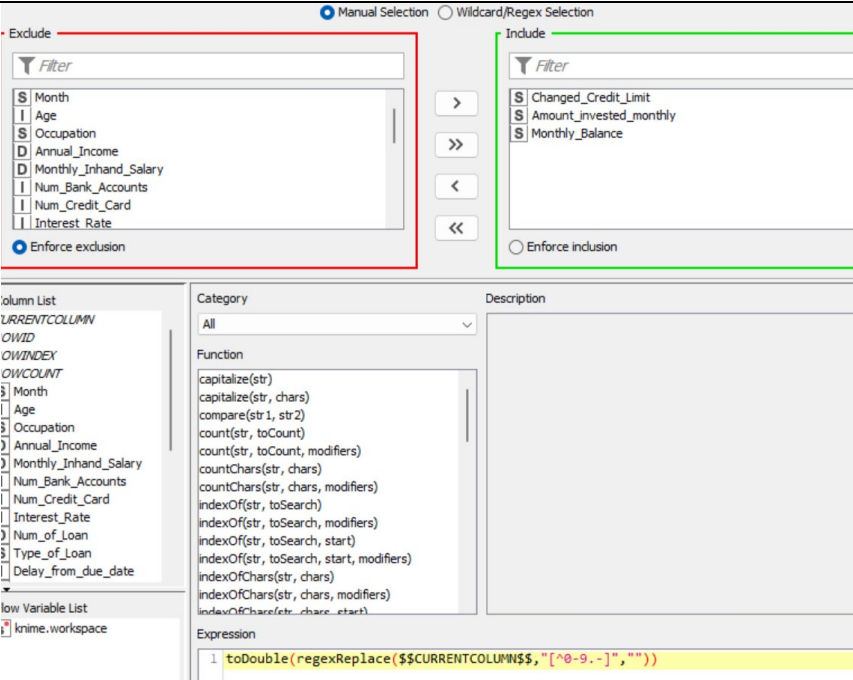

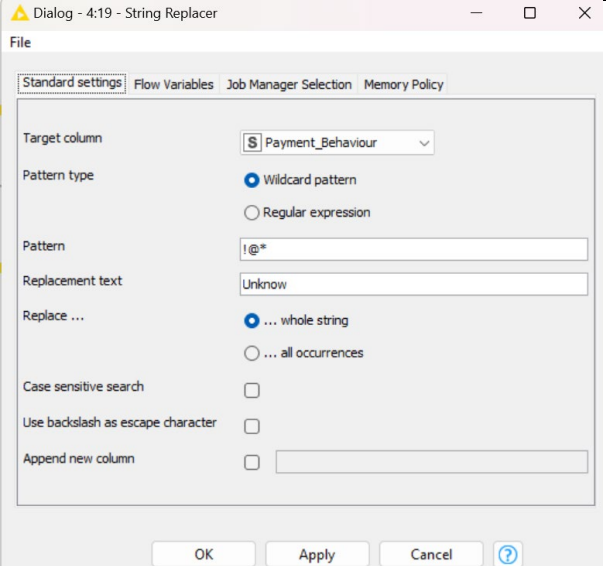
Ans:

Sequence	Node	Command
1	String Manipulation	<div> <div>Expression</div> <div> <pre>1 toInt(strip(substr(\$Credit_History_Age\$,0,2)))*12+toInt(strip(substr(\$Credit_History_Age\$,indexOf(\$Credit_History_Age\$, "d")+1,3)))</pre> </div> <div>command:</div> </div> <p>toInt(strip(substr(\$Credit_History_Age\$,0,2)))*12+toInt(strip(substr(\$Credit_History_Age\$,indexOf(\$Credit_History_Age\$, "d")+1,3)))</p> <p>String Manipulation</p> <p>1. explain:</p> <p>substr(\$Credit_History_Age\$, 0, 2):</p> <ul style="list-style-type: none"> Extracts the first two characters from Credit_History_Age, which usually represents the number of years (e.g., "5 " from "5 Years and 3 Months"). <p>2. strip(...):</p> <ul style="list-style-type: none"> Removes any extra spaces around the extracted substring for clean conversion. <p>3. toInt(...) * 12:</p> <ul style="list-style-type: none"> Converts the cleaned year substring into an integer and multiplies it by 12 to get the total months for the years. <p>4. indexOf(\$Credit_History_Age\$, "d") + 1:</p> <ul style="list-style-type: none"> Finds the position of the character "d" in the word "and" within Credit_History_Age (e.g., in "Years and "). Adding 1 moves to the starting position of the month portion. <p>5. substr(..., indexOf(...) + 1, 3):</p> <ul style="list-style-type: none"> Extracts up to three characters after "and", typically representing the number of months (e.g., " 3 " in "5 Years and 3 Months"). <p>6. toInt(strip(...)):</p> <ul style="list-style-type: none"> Cleans and converts the extracted month substring to an integer. <p>7. Total Calculation:</p> <ul style="list-style-type: none"> Adds the months from years (years * 12) to the extracted month value to get the final count in months. <p>Example</p> <p>For "5 Years and 3 Months", the expression would calculate:</p> <ul style="list-style-type: none"> Years part: $5 * 12 = 60$ Months part: 3 Total: $60 + 3 = 63$ months.

- 7) Remove the non-numerical symbol in “Amount_invested_monthly” and convert it to the double format. Set the value to “Unknow” if the original value in “Payment_Behaviour” attribute starts with “!@”. Remove the non-numerical symbol in “Monthly_Balance” and convert it to the double format. Convert “Changed_Credit_Limit” into the double format. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[5 marks]**

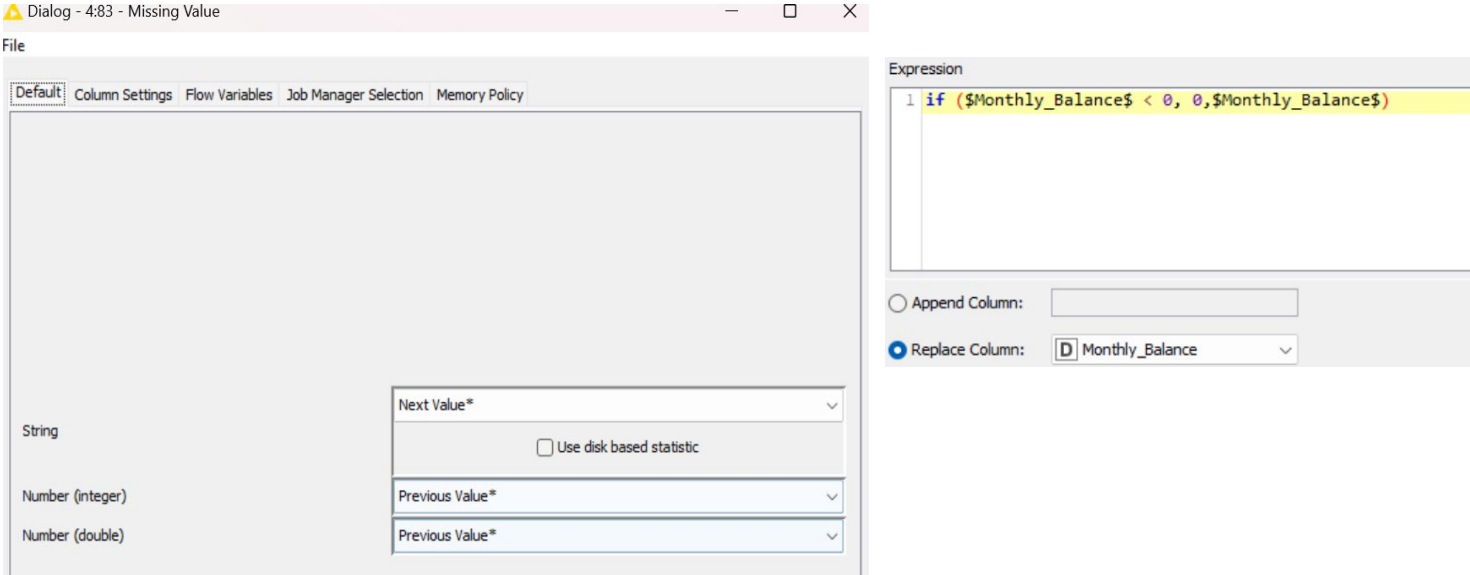
Ans:

Sequence	Node	Command
----------	------	---------

<p>1</p>	<p>String Manipulation (Multi-Column)</p> 	 <p>Remove the non-numerical symbol in “Amount_invested_monthly”, “Monthly_Balance”, “Changed_Credit_Limit” and convert into the double format</p>
<p>2</p>	<p>String Replacer</p> 	 <p>Use a wildcard pattern to find and replace any string that begins with “!@”.</p>

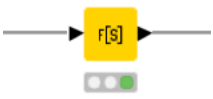
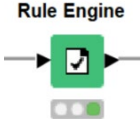
- 8) Use the “Missing Value” node and use the “Next Value*” to replace missing values in all string type attributes. Use the “Previous Value*” in the same node to replace missing values in any numerical format. If the value of “Monthly_Balance” is negative, replace the value with 0. Screenshot the pop-up window with the correct settings. **[5 marks]**

Ans:
Here are the screenshots of the "Missing Value" and "Math Formula" nodes. In the "Math Formula" node, we use an if statement to check if the value is less than 0. If the value is negative, it is changed to 0; otherwise, the value remains unchanged.



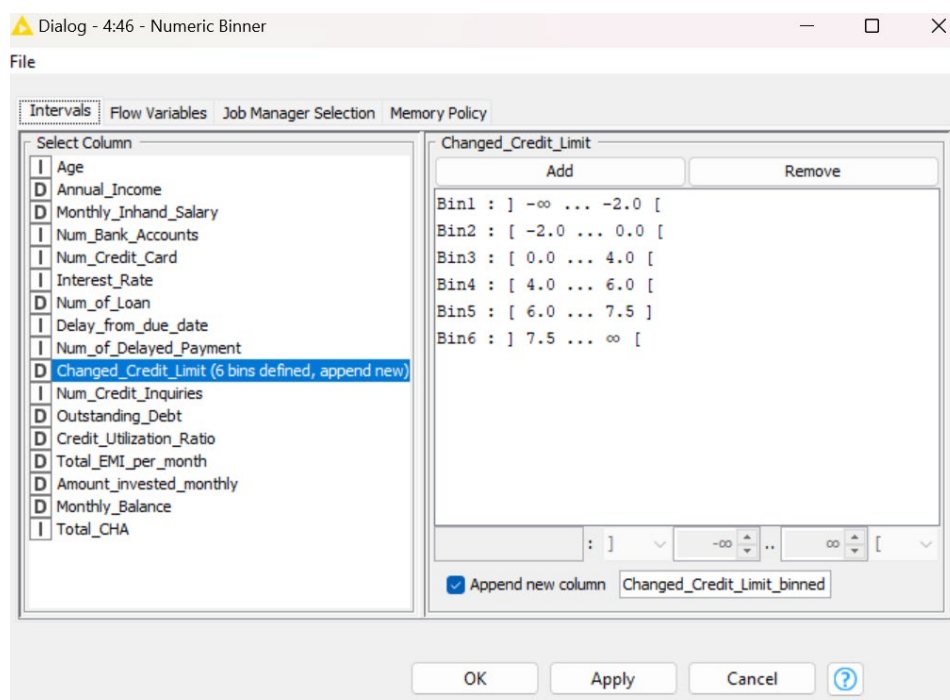
- 9) Simplify the “Type_of_Loan” attribute. If the original content has more than one type separated by a comma, keep only the first part. Otherwise, keep the full description if there is no comma included. For example, “Auto Loan, Credit-Builder Loan, Personal Loan, and Home Equity Loan” will become “Auto Loan”, “Credit-Builder Loan” will still be “Credit-Builder Loan”, and “Not Specified, Auto Loan, and Student Loan” will become “Not Specified” after the process. List the node(s) (in sequence) and the corresponding command(s) used in this process. **[10 marks]**

Sequenc e	Node	Command
1	String Manipulation <div> String Manipulation <div> r[s] </div> </div>	<div> <div> Expression <div> 1 string(\$Type_of_Loan\$) </div> </div> <div> Append Column: new_loan Replace Column: Total_CHA </div> <div> Insert Missing As Null Syntax check on close </div> </div> <div> Converts the value of \$Type_of_Loan\$ into a string to ensure it's treated as a string type. The command tick append columns new_loan appends a new column called new_loan to store the derived data. </div>

2	String Manipulation		<p>Expression</p> <pre>1 substr(\$Type_of_Loan\$,0,indexOf(\$Type_of_Loan\$,","))</pre> <p> <input type="radio"/> Append Column: <input type="text"/> <input checked="" type="radio"/> Replace Column: <input type="text" value="\$ Type_of_Loan"/> </p> <p> <input type="checkbox"/> Insert Missing As Null <input checked="" type="checkbox"/> Syntax check on close </p> <p>Extracts the substring from \$Type_of_Loan\$, capturing everything before the first comma by finding its position with indexOf and using substr to slice the string accordingly. Then, replaces the original Type_of_Loan column with this extracted substring</p>
3	Rule Engine		<p>Expression</p> <pre>? 1 // enter ordered set of rules, e.g.: ? 2 // \$double column name\$ > 5.0 => "large" ? 3 // \$string column name\$ LIKE "*blue*" => "small and blue" ? 4 // TRUE => "default outcome" S 5 \$Type_of_Loan\$ LIKE "" => \$new_loan\$ S 6 TRUE => \$Type_of_Loan\$</pre> <p> <input type="radio"/> Append Column: <input type="text" value="prediction"/> <input type="text" value="S"/> <input checked="" type="radio"/> Replace Column: <input type="text" value="\$ Type_of_Loan"/> </p> <p>The rule checks if \$Type_of_Loan\$ is empty, and if true, it replaces it with the value from new_loan. The rule TRUE => \$Type_of_Loan\$ ensures that if no condition is met, the original \$Type_of_Loan\$ value remains unchanged. Finally, replace column Type_of_Loan updates the Type_of_Loan column with the modified values based on these rules.</p>

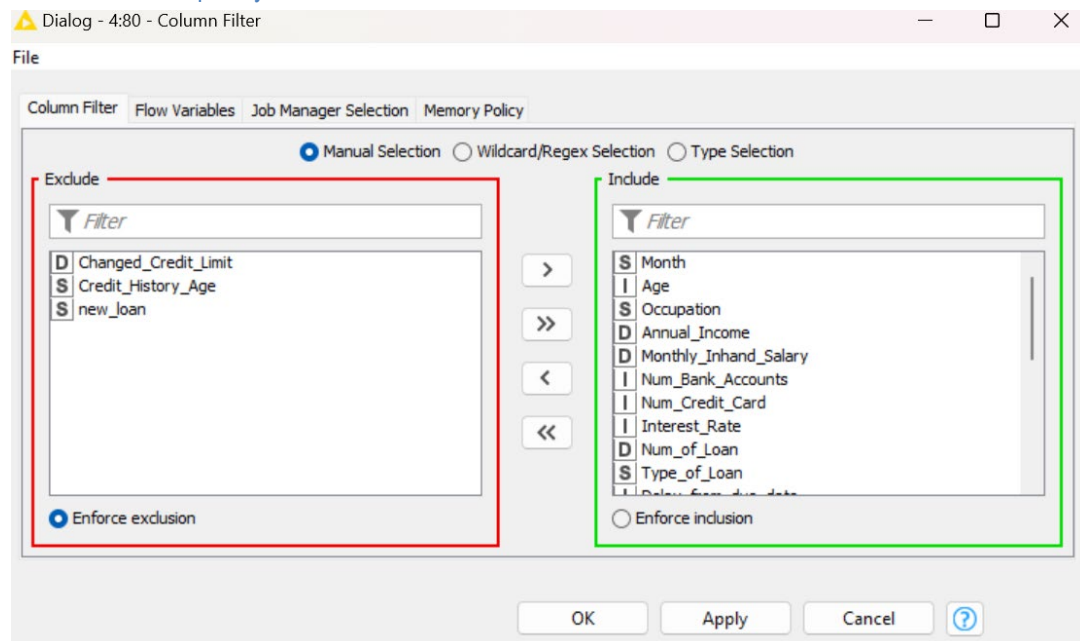
- 10) Bin the "Changed_Credit_Limit" attribute with six bins of ranges: $[-\infty, -2.0)$, $[-2.0, 0)$, $[0, 4.0)$, $[4.0, 6.0)$, $[6.0, 7.5)$, and $[7.5, \infty)$ and put the result into a new attribute called "Changed_Credit_Limit_binned". Screenshot the pop-up window with the correct settings of your binner. **[5 marks]**

Ans:

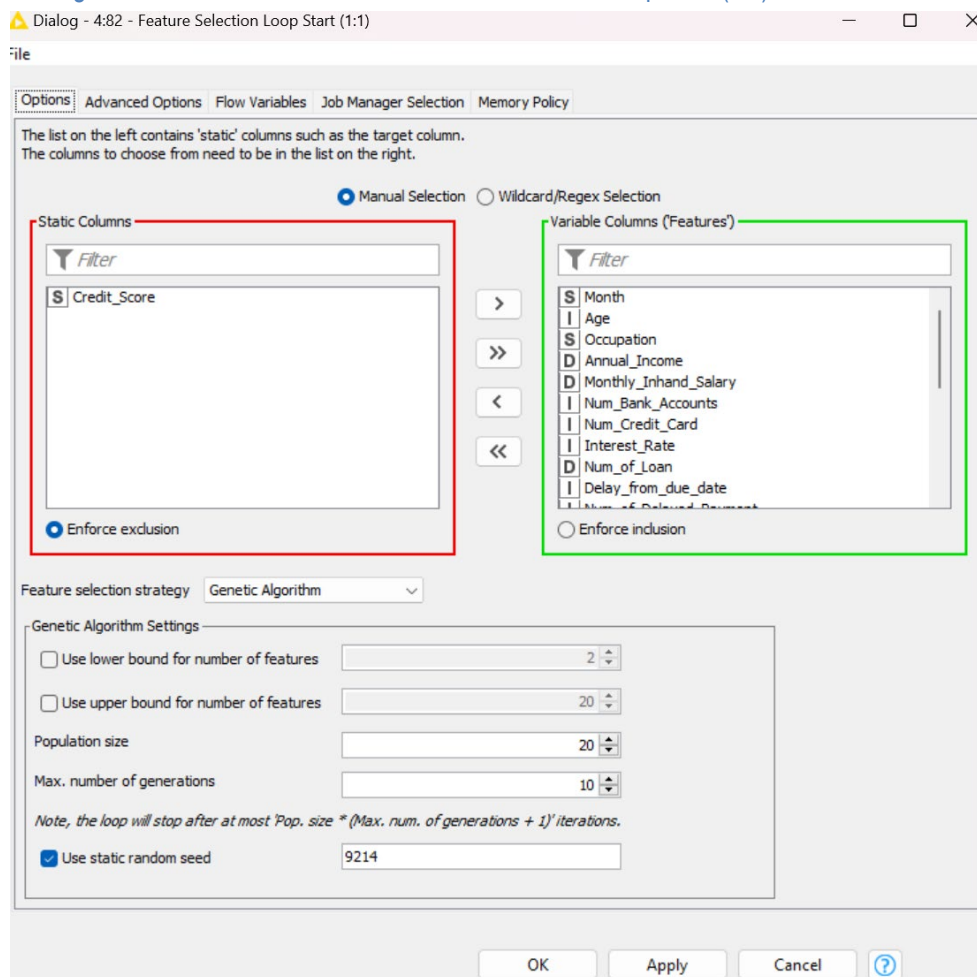


- 11) Remove all temporarily created or useless attributes. Use the “Feature Selection Loop Start (1:1)” node to select the feature. The class label should be excluded from the features in the feature selection node. The Genetic Algorithm is specified to be the feature selection strategy with default population size and the maximum number of generations. Again, **9214** should be used as the static random seed. After selecting features, shuffle the data with seed **9214**. The data should be partitioned by “Linear sampling”, with 80% data in the training set and 20% in the test set. How many tuples and attributes (excluding the class label) are in the training set at the end? **[5 marks]**

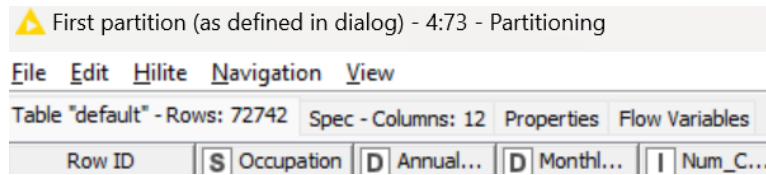
Ans: Remove temporary class with Column Filter:



Removing the "Credit Score" attribute in Feature Selection Loop Start (1:1) because it serves as a label attribute.



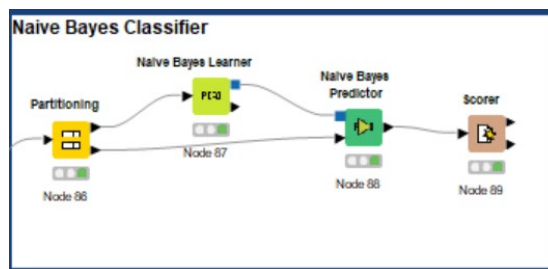
There are 72742 tuples and 12 attributes in training set at the end.



2. Build a Naïve Bayes classifier using the training and test sets created in the previous task. Answer the following questions after completing the model training and test. **[15 marks in total]**

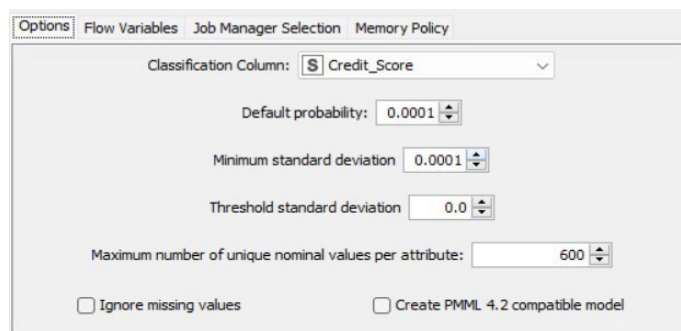
- 1) Give a screenshot of the Naïve Bayes classifier in the KNIME workflow. You can take the screenshot starting from the portioning node output to the end of the Naïve Bayes classifier part scorer. **[2.5 marks]**

Ans:



- 2) The default probability should be 0.0001, the minimum standard deviation is 0.0001, the threshold standard deviation is 0, and the maximum number of unique nominal values per attribute should be set to 600 in the classifier. Screenshot the setting dialogue of your Naïve Bayes Learner. **[2.5 marks]**

Ans:



- 3) Screenshot the **confusion matrix** and the **Accuracy statistics** of the test result. If the bank wants to minimise the risk of lending money to customers, the "Good" in "Credit_Score" should be the major target. Based on the current result, does the classifier perform satisfactorily? **[5 marks]**

Ans:

Credit_Sco...	Good	Standard	Poor
Good	2632	588	65
Standard	2055	5847	1731
Poor	783	1670	2815

Correct classified: 11,294 Wrong classified: 6,892
Accuracy: 62.103% Error: 37.897%
Cohen's kappa (κ): 0.404%

Row ID	TruePo...	FalsePo...	TrueNe...	FalseN...	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
Good	2632	2838	12063	653	0.801	0.481	0.801	0.81	0.601	?	?
Standard	5847	2258	6295	3786	0.607	0.721	0.607	0.736	0.659	?	?
Poor	2815	1796	11122	2453	0.534	0.61	0.534	0.861	0.57	?	?
Overall	?	?	?	?	?	?	?	?	?	0.621	0.404

Analysis:

- **Accuracy:** The model correctly classifies 62% of instances, but this isn't sufficient for minimizing lending risk, especially given class imbalances.
- **Cohen's Kappa:** A moderate value of 0.404 indicates that the model's predictions and the actual labels are not strongly aligned.
- **"Good" Class:**
 - ☐ Recall (0.801) indicates that the model correctly identifies 80.1% of "Good" customers, but **Precision (0.481)** is low, meaning the model misclassifies a significant number of "Standard" or "Poor" customers as "Good," which increases the risk for the bank.
 - ☐ The **False Negatives** (588) for the "Good" class show that the model is missing some "Good" customers, which could result in missed opportunities for the bank.

Conclusion:

The classifier does not meet the bank's goal of minimizing lending risk. While the overall accuracy is 62.1%, the low Precision and Recall for the "Good" class, along with False Negatives, suggest it misses trustworthy customers, posing a risk for the bank. Improvements in Precision and Recall for the "Good" class are needed.

4) Which measurement should we look at to interpret your conclusion in this case? [5 marks]

Ans:

To evaluate the classifier's effectiveness in minimizing lending risk by accurately identifying "Good" credit customers, the most important metric to consider is **Precision** for the "Good" class.

Precision is crucial because it measures how accurately the classifier labels customers as "Good" without misclassifying higher-risk customers ("Standard" or "Poor") as "Good." A high precision reduces false positives, which is essential for minimizing lending risk and avoiding loans to high-risk customers. Therefore, Precision is the key measure of the classifier's performance in this context.

In the confusion matrix, **Precision** is calculated as:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Where:

- **True Positives (TP)** is the number of "Good" customers correctly identified.
- **False Positives (FP)** is the number of "Standard" or "Poor" customers incorrectly identified as "Good."

A high precision value indicates that the model avoids misclassifying "Standard" or "Poor" customers as "Good," which is critical to minimizing lending risk. However, in this case, the classifier shows a **low precision of 0.481** for the "Good" class, meaning it frequently misclassifies higher-risk customers as "Good." This indicates that the classifier's performance is not satisfactory for the bank's risk minimization goals.

Additional Metrics:

- **Recall (Sensitivity):** Measures how well the model captures all true "Good" customers. However, it doesn't account for false positives, which are vital for lending risk assessment.
- **Specificity:** Indicates how well the model identifies "Standard" or "Poor" customers. While useful, it is not directly relevant to identifying "Good" customers.
- **F1 Score:** Balances precision and recall but is less focused on false positives, which is the main concern for minimizing risk in this case.

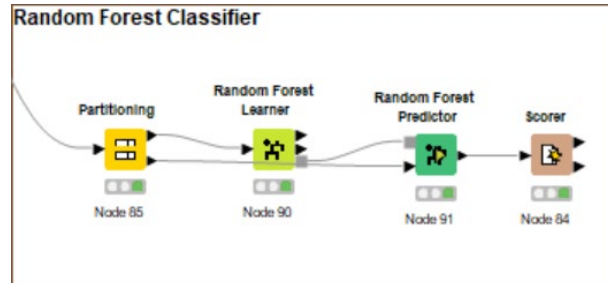
Conclusion:

Precision for the "Good" class directly addresses the bank's goal of minimizing lending risk. The low precision in this classifier suggests that it is misclassifying too many higher-risk customers as "Good," which makes it unsuitable for the bank's lending decisions.

3. Build a random forest classifier using the training and test sets created in the previous task. Answer the following questions after completing the model training and test. Use the information gain ratio as the split criterion and **9214** as the static random seed to build the random forest model. **[15 marks in total]**

- 1) Give a screenshot of the random forest classifier in the KNIME workflow. You can take the screenshot starting from the partitioning node output to the end of the Naïve Bayes classifier part scorer. **[2.5 marks]**

Ans:



- 2) Screenshot the confusion matrix and the Accuracy statistics of the test result. **[2.5 marks]**

Ans:

Confusion Matrix - 4:79 - Scorer

Credit_Sco...	Good	Standard	Poor
Good	2323	933	29
Standard	792	7639	1202
Poor	92	975	4201

Correct classified: 14,163 Wrong classified: 4,023
 Accuracy: 77.879% Error: 22.121%
 Cohen's kappa (κ): 0.633%

Accuracy statistics - 4:79 - Scorer

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's...
Good	2323	884	14017	962	0.707	0.724	0.707	0.941	0.716	?	?
Standard	7639	1908	6645	1994	0.793	0.8	0.793	0.777	0.797	?	?
Poor	4201	1231	11687	1067	0.797	0.773	0.797	0.905	0.785	?	?
Overall	?	?	?	?	?	?	?	?	?	0.779	0.633

- 3) If the bank wants to minimise the risk of lending money to customers, the “Good” in “Credit_Score” should be the major target. Compare the measurements between random forest results and Naïve Bayes results. Which model presents a more suitable result? Which measure should be used to make the comparison? **[5 marks]**

Ans:

	Naïve Bayes model	Random Forest model
Precision (Good)	0.481	0.724
Accuracy	0.621	0.779

If the bank wants to minimize the risk of lending money to customers, the “Good” in “Credit_Score” should be the major target. Compare the measurements between Random Forest results and Naïve Bayes results. Which model presents a more suitable result? Which measure should be used to make the comparison? **[5 marks]**

Answer:

To minimize the risk of lending, the bank should focus on reducing the **False Positive Rate (FPR)** for the "Good" class in Credit_Score. A low FPR for the "Good" class means the model is less likely to misclassify risky customers (Standard or Poor) as Good, which is critical to reducing lending risk.

Calculation of False Positive Rate (FPR) for "Good" Class

- False Positives (FP) for Good** = 792 (Standard misclassified as Good) + 92 (Poor misclassified as Good) = 884
- True Negatives (TN) for Good** = 14,017 (Total of Standard and Poor correctly not classified as Good)
- FPR (Type I Error) for Good:**

$$FPR = \frac{FP}{FP + TN} = \frac{884}{884 + 14017} = 0.0593 \text{ or } 5.93\%$$

Classifier	False Positive Rate (FPR) for "Good"
Naïve Bayes	19.04%
Random Forest	5.93%

Comparison:

- The **Random Forest Classifier** has a significantly lower FPR of **5.93%** compared to Naïve Bayes' FPR of **19.04%** for the "Good" class. This indicates that Random Forest is less likely to incorrectly classify risky customers as Good, making it the more suitable model for minimizing lending risk.

Conclusion: The **False Positive Rate (FPR)** is the most appropriate measurement for this comparison, as it shows which model has fewer Type I errors when classifying "Risky" (Standard & Poor) as "Non-Risky" (Good). Based on FPR, the **Random Forest model is more suitable** as it minimizes the risk of lending to potentially risky customers.

- 4) Which class does the built random forest model perform the best? What measurement(s) should we look at to find the answer? **[5 marks]**

Ans:

To determine the class where the Random Forest model performs best, we should examine the **False Positive Rate (FPR)** for each class. The class with the lowest FPR has the highest accuracy in identifying only true positives without mistakenly including instances from other classes.

FPR Calculation for Each Class

Class	False Positive Rate (FPR)
Good	5.26%
Standard	22.37%
Poor	8.23%

Conclusion

The **"Good" class** has the lowest FPR at **5.26%**, meaning the model has a low rate of incorrectly classifying risky customers (Standard and Poor) as Good. This indicates that the Random Forest model is most effective in identifying the "Good" class accurately, making it the best-performing class for minimizing credit risk.

----- End of Submission -----