**Cover sheet for submission of**

**work for assessment**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UNIT DETAILS** | | | | | | | | | | |
| Unit name | | Data Science Principles | | | | | | Class day/time | Friday | Office use only |
| Unit code | | COS10022 | | | Assignment no. | | 2 | Due date | 12/Nov/2023 |  |
| Name of lecturer/teacher | | | | Huy Truong | | | | | |  |
| Tutor/marker’s name | | | Huy Truong | | | | | | | Faculty or school date stamp |
| **STUDENT(S)** | | | | | | | | | | |
| Family Name(s) | | | | | | Given Name(s) | | | | Student ID Number(s) |
| (1) | Nguyen Dinh | | | | | Nhat Minh | | | | 103802490 |

|  |  |  |
| --- | --- | --- |
| **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:   1. 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) |  |  |

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/** PAGE 1 OF 1

# http://www.swinburne.edu.my/images/2013/logo.jpgSwinburne University of Technology Hawthorn Campus Dept. of Computer Science and Software Engineering

**COS10022 Data Science Principles**

Assignment 2 - *Semester3 2023*

**Assessment Title**: Predictive Model Creation and Evaluation

## Assessment Weighting: 30%

**Due Date**: Saturday, 12th November 2023 at 11.59 pm (GMT+7)

**Assessable Item:**

* One (1) piece of a written report no more than 10-page long with the signed Assignment Cover Sheet.
* A unit peer must review your submission before it can be marked.

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 **3122** to the seed to fix the random state. **[70 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. Exclude the attribute(s) and give a persuasive rationale for that. The excluded attribute(s) is(are) , and the reason(s) for removing it(them) is(are) . **[5 marks]**

Ans:

Since these attributes are needed for later task, I would clean the raw data first and then remove the selected attributes just before partitioning and importing into the models.

|  |  |
| --- | --- |
| **Removed Attributes** | **Reason** |
| Month | We do not rely on the period to determine our credit score, hence this information is redundant and should be remove. |
| Name | Since our primary focus is determining credit scores, a secondary identifier is unnecessary. Additionally, using personal information can pose privacy concerns. |
| Age | Using personal information to assess creditworthiness is often illegal. |
| Occupation | Although this attribute can be related to loan repayment ability and credit improvement, it is less significant compared to income or credit history age. |
| Type\_of\_Loan | Similar to occupation, while the types of loans may demonstrate one's ability to repay loans, is not necessarily more significant than, for example, the number of loans. |
| Payment\_Behaviour | The number of delayed payments and payment due dates is a more decisive factor, making this attribute somewhat redundant |

.

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nodes** | **Command** | **Explanation** |
| 1 |  |  | To make sure the defined rows that has missing value are removed, we use the ‘Missing Value’ nodes with the ‘Remove Row’ configuration. |
| 3 |  | $Monthly\_Inhand\_Salary$ **<** **0** => **TRUE**  $Num\_Bank\_Accounts$ **<** **0** => **TRUE**  $Num\_Credit\_Card$ **<** **0** => **TRUE**  $Changed\_Credit\_Limit$ **MATCHES "[^0-9.-]"** => **TRUE**  ‘*Exclude TRUE matches’ option choose* | Using the defined condition, we are able to roughly clean the data and complete the task at hand. |

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nodes** | **Command** | **Explanation** |
| 1 |  | toInt**(**regexReplace**(**$Age$,**"[^0-9.-]"**,**""))** | Since we would be recovering the values that should be numeric but aren’t. In this case, we would use regular expression to replace non-numerical characters with an empty character, effectively removing the non-numerical characters. |
| 2 |  | $Age\_Cleaned$ **<= 0 AND** $Age\_Cleaned$ **>=** 120 => **TRUE**  ‘*Exclude TRUE matches’ option choose* | Setting the parameters to eliminate the ages outside of 0 and 120. |

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  | toDouble**(**regexReplace**(**$Annual\_Income$,**"[^0-9.-]"**,**""))** | We would use regular expression to replace non-numerical characters with an empty character, effectively removing the non-numerical characters. |

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  | toNull**(**replace**(**$Occupation$, **"\_\_\_\_\_\_\_"**,**""))** | Replace the defined attribute to an empty string, then use toNull function to replace empty string with null value. |
| 2 |  | toInt**(**regexReplace**(**$$CURRENTCOLUMN$$,**"[^0-9.-]"**,**""))** | Replace non-numerical character with empty string, then convert the type of the selected column to integer. And since both “Num\_of\_Loan” and “Num\_of\_Delayed\_Payment” has the same requirement, we use he multi column option of ‘String Manipulator’ to do our bidding. |
| 3 |  | if**(**$Num\_of\_Loan$**<0**,**0**,$Num\_of\_Loan$**)** | Replace the integer with value < 0 with a 0 after converting ‘$Num\_of\_Loan$’ to integer. |
| 4 |  | abs**(**$$CURRENT\_COLUMN$$**)** | Since these 2 column has the same requirement, we use the ‘Math Formular (Multi Column)’ to get the absolute value of both of them at the same time. |
| 5 |  | replace**(**$Credit\_Mix$, **"\_"**, "Unknown") | Replace the defined parameter with “Unknown” |
| 6 |  | toDouble**(**regexReplace**(**$Outstanding\_Debt$,**"[^0-9.-]"**,**""))** | Replace non-numerical character with empty string, then convert the type of the selected column to double |

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  | toInt**(**strip**(**substr**(**$Credit\_History\_Age$,**0**,**2)))** | This node handles the “Credit\_History\_Age” by first getting the first 2 digit in the string. Then trim all the trailing space where applicable. Finally convert the value to Integer. We append the new column named “CHA\_Year” to store the value of this operation |
| 2 |  | toInt**(**strip**(**substr**(**$Credit\_History\_Age$,indexOf**(**$Credit\_History\_Age$, **"d")** **+ 1**,**3)))** | The same treatment is used for this node, with the addition of getting the position after the letter “d” to act as the starting point of the substring.  We use +1 because we want to not get the letter “d” into our result. We append the new column named “CHA\_Month” to store the value of this operation |
| 3 |  | $CHA\_Year$ **\* 12** **+** $CHA\_Month$ | Multiply the year by 12 and add the month to get the total CHA. |

* 1. 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:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  | toDouble**(**regexReplace**(**$$CURRENTCOLUMN$$,**"[^0-9.-]"**,**""))** | Since “Amount\_invested\_monthly”, “Payment\_Behaviour”, and “Changed\_Credit\_Limit” has similar condition, we would use one multi Column string manipulator to handles them.  Even though “Changed\_Credit\_Limit” does not need non-numerical character removed. We would include it to make the most out of the to Double function, beside it add an extra layer of cleansing. |
| 2 |  |  | We use ‘String Replacer’ node to change all occurrence of “!@” in the “Payment\_Behaviour” column. And since we want to replace the starting pattern, we put an asterisk at the end to get the value after the defined pattern. |

* 1. 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:

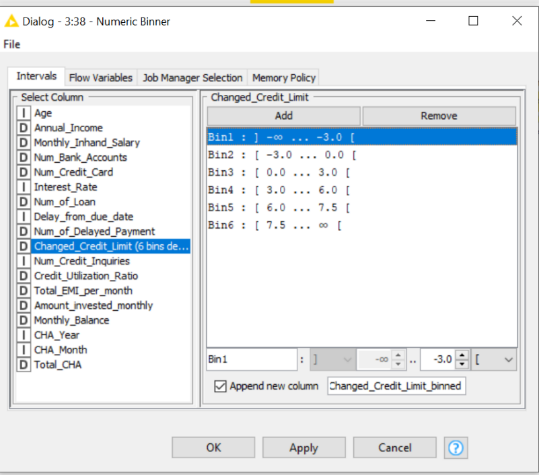
|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  |  | Configured according to the task requirement. |
| 2 |  | **if(**$Monthly\_Balance$**<0**,**0**,$Monthly\_Balance$**)** | Replace the number with value < 0 with a 0. If the condition false, the value remain the same. |

* 1. 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]**

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Node** | **Command** | **Explanation** |
| 1 |  | substr**(**$Type\_of\_Loan$, **0**, indexOf**(**$Type\_of\_Loan$, **"**,**"))** | In this command, we get the position of the “,” first, this will also act as our string length. Then we get the subtring starting from the 0 index position to get the first description of the “Type\_of\_Loan”.  We store the handled value under a new column named “ToL\_New”.  Empty value can be present after this step, which is why we would need to use another node. |
| 2 |  | $ToL\_New$ **LIKE** **""** => $Type\_of\_Loan$  TRUE => $ToL\_New$ | This node replace the empty string in the “ToL\_New” column with the value in the original “Type\_of\_Loan” column. If the string is not empty, it remains the new value. |

* 1. Bin the “Changed\_Credit\_Limit” attribute with six bins of ranges: , , , , , and 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]**

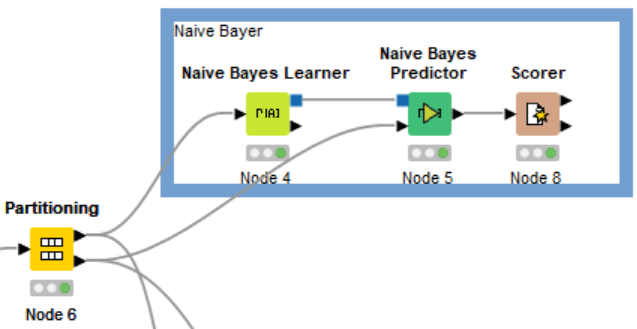
Ans:

* 1. 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, **3122** should be used as the static random seed. After selecting features, shuffle the data with seed **3122**. The data should be partitioned by “Linear sampling”, with 75% data in the training set and 25% in the test set. How many tuples and attributes (excluding the class label) are in the training set at the end? **[5 marks]**

Ans:

There are **68196 tuples and 10 attributes** in training set at the end.

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

1. 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]**

A screenshot of a computer

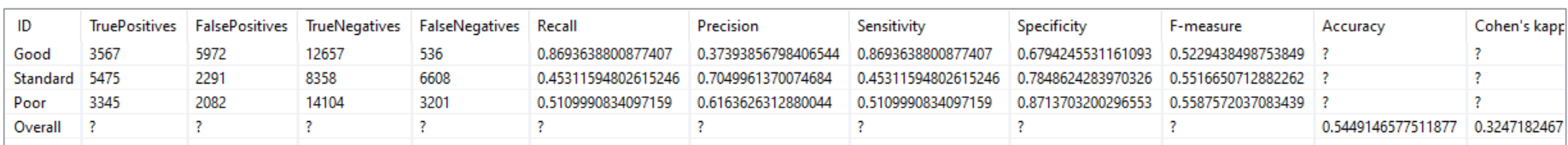
Description automatically generatedAns:

1. 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:

**A screenshot of a computer

Description automatically generatedConfusion matrix:**

**Accuracy statistic:**

From the result of the Naïve Bayes model, we can say that the classifier does not perform very well when it comes to identifying good credit since the precision for identifying “Good” credit is rather low. Hence, we can’t say that it satisfies the criteria of minimising lending risk for banks.

1. Which measurement should we look at to interpret your conclusion in this case? **[5 marks]**

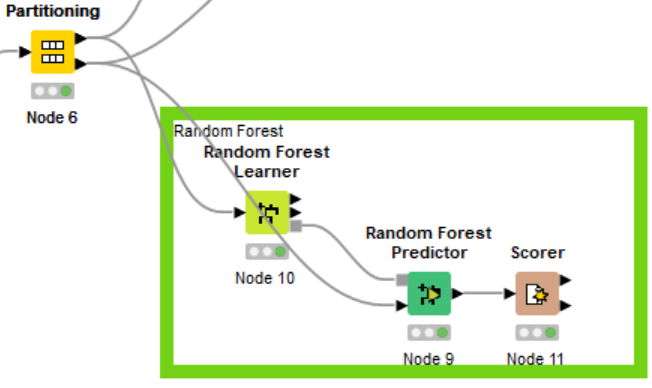
Ans:

In this model, our primary focus is on the "Good" class because it serves as the most reliable determinant for identifying eligible borrowers with minimal risk for the bank. While the precision rate for this class may be relatively low, it remains the key factor in assessing who can safely borrow money.

This approach is reasonable because individuals with "Standard" or "Poor" credit are less likely to repay the money to the bank on time or even at all. Lending to such individuals increases the risk of the bank losing money. On the other hand, selecting accounts with "Good" credit credibility implies a higher likelihood of timely repayment, thereby reducing the risk of financial loss for the bank. By accurately identifying and approving accounts with a "Good" credit status, the bank can significantly enhance its safety and confidence in minimizing the lending risk.

On the other hand, even though the recall rate for the “Good” class is high, we cannot rely on it as it dismissed all the case of “Standard” or “Poor” that are wrongly determined as “Good”. As we want to have an accurate vision, relying on potentially misleading numbers can prove disastrous. Regarding F-measure, it is a combination of both recall and precision rate, hence, it still confines the risk of generalizing the data and provide inaccurate calculations to determine lending risks.

1. Build a random forest classifier using thetraining 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 **3122** as the static random seed to build the random forest model. **[15 marks in total]**
2. Give a screenshot of the random forest 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]**

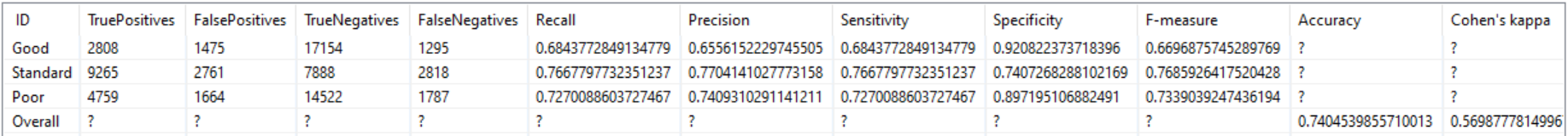
Ans:

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

Ans:

**A screenshot of a computer

Description automatically generatedConfusion matrix:**

**Accuracy statistic:**

It's evident that the Random Forest model outperforms the Naïve Bayes model in this comparison. The Random Forest model demonstrates a higher overall accuracy of over 70%. In contrast, the Naïve Bayes model achieves just over 50% accuracy, meaning it correctly predicts a lower percentage of the classes.

1. 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:

As stated in task 2.4 , we would rely on the precision rate, as the recall and F-measures only give an overall view and not the accurate result banks would want in the decision making process of lending money.

|  |  |  |
| --- | --- | --- |
| **Naïve Bayes** | **Random Forest** | **Measurement** |
| 0.3739 | 0.6556 | Precision |

Comparing the precision rate between the model side by side, we can clearly see that the Random Forest model performs much better than Naïve Bayes, this is due to the fact that Random Forest is generally superior to Naïve Bayes in determining account credibility due to its ability to model complex relationships, handle irrelevant features, manage imbalanced data, mitigate overfitting, and provide higher accuracy, making it a more suitable choice for tasks like credit scoring where precision is crucial for minimizing risk.

1. 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:

Performance in this case would be measured by the high precision of the model in picture.

|  |  |  |
| --- | --- | --- |
| **Class** | **Random Forest** | **Measurement** |
| Good | 0.6556 | Precision |
| Standard | 0.7704 |
| Poor | 0.7409 |

Oberserving the precision of the Random Forest model, we can see that the “Standard” Class has the highest precision rate between the 3 classes. Although the standard class wouldn’t be considered in minimizing lending risk, its high precision still proves that for complex model, Random Forest is the best for determining accuracy, but it does come at the cost of being rather slow and resource-intensive.

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