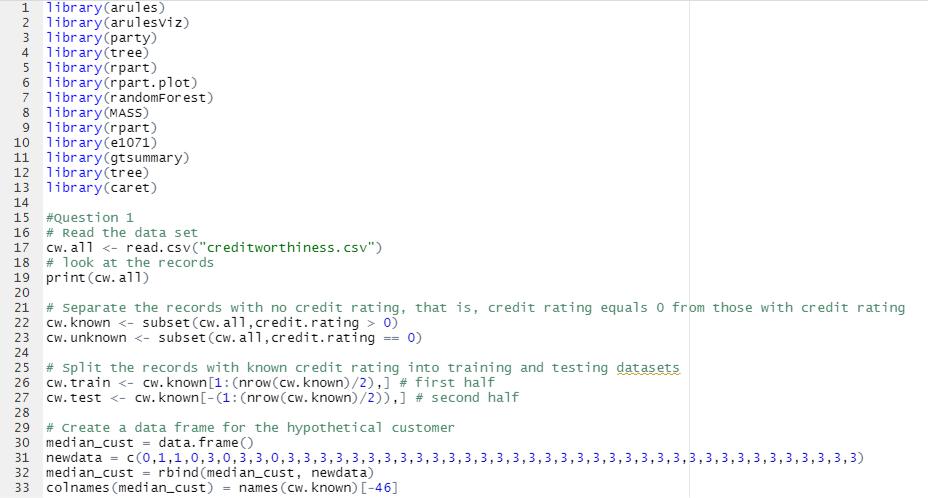
**Individual Assignment 2**

**Name: Zhang XueCheng**

**Student ID: 7020739**

**Question 1:**



For question 1, load the needed libraries and read in the data set “creditworthiness.csv”.

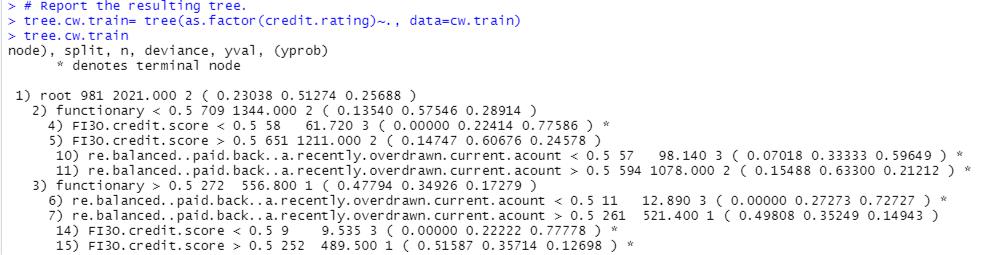
To predict the credit rating that would be assigned to each individual, we need to first separate the records and use those records with credit rating > 0. I have name it as cw.known.

After splitting the records with cw.known, we will split the dataset into a training and a test set.

**Question 2:**

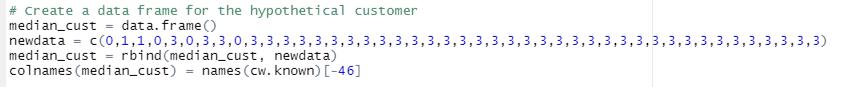
a)

Make use of the tree package **library(tree)** to report the resulting tree



b)

Based on the resulting tree output “tree.cw.train”, we can predict the credit rating of a hypothetical “median” customer



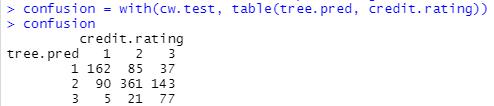
We first create a data frame for the hypothetical customer, and then insert some data inside, then use the row-bind function rbind to combine the data frame and new data together, and the column name will be the same as cw.known.



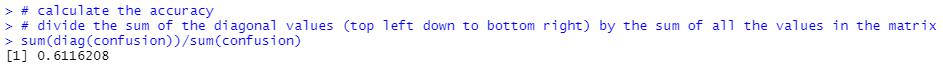
This will give the levels result output

c)

After that, we can start to produce the confusion matrix for predicting the credit rating from the tree produce earlier



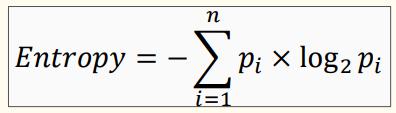
After we generated the confusion matrix, we can calculate the accuracy

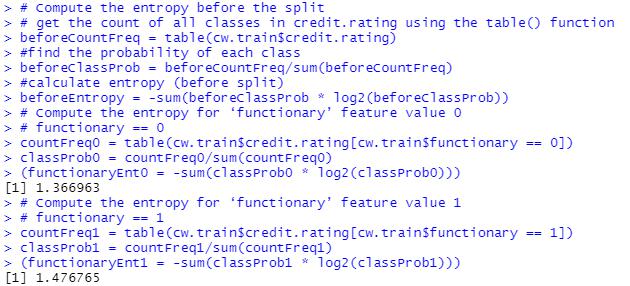


The accuracy rate is 61.2%

d)

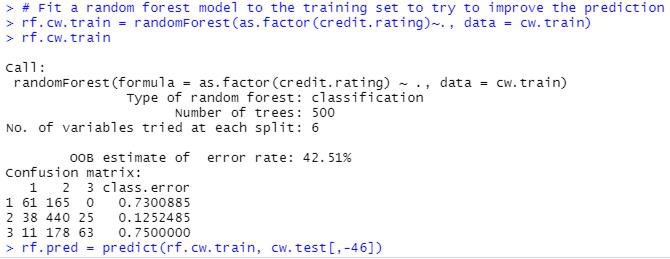
To calculate the numerical value of the gain in entropy corresponding to the first split at the top of the tree, we first need to compute the entropy before the split by using the below formula:



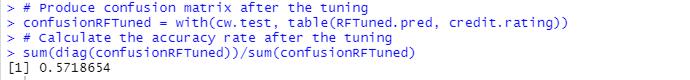


e)

Fit a random forest model to the training set to try to improve prediction



f)



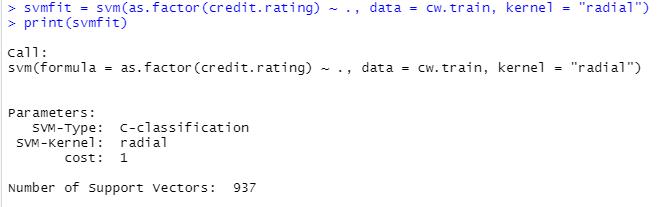
With the prediction value, we can produce the confusion matrix between the test dataset and predicted value. Since the overall accuracy is 57.2% and only decreased by 4.0% compared to the accuracy without tunning, hence overfitting does not occur.

**Question 3:**

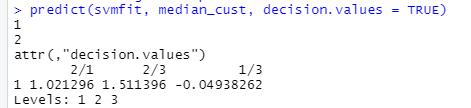
By using default settings for svm() from the e1071 package, we can fit a support vector machine to predict the credit ratings of customers using all of the other variables in the dataset.

a)

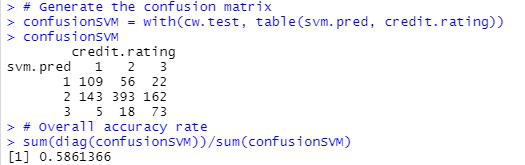
Predict the credit rating of a hypothetical “median” customer



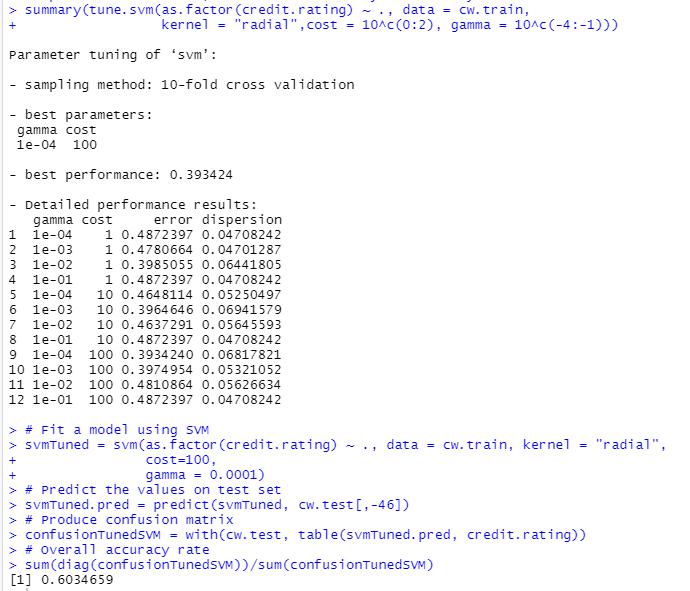
Report the decision value:



b)



c)

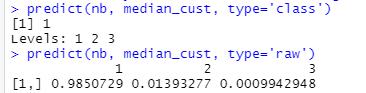


**Question 4:**

Fit the Naive Bayes model to predict the credit ratings of customers using all of the other variables in the dataset.

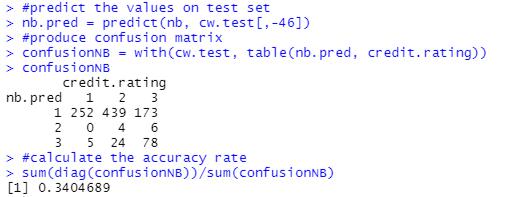


a)



b)

We can use the Naïve Bayes model to predict, produce the confusion matrix, and then compute the accuracy rate.



**Question 5:**

a) Which of the classifiers look to be the best?

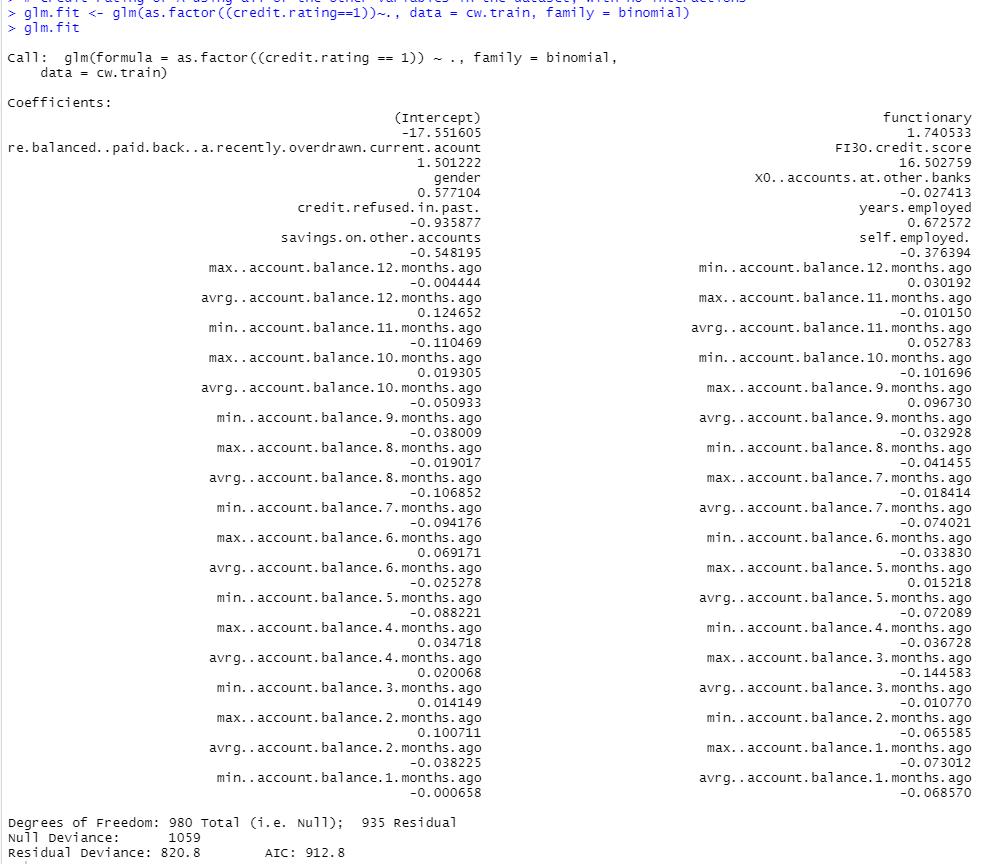
As we can see from the above results, decision tree classifier gives an overall accuracy of 61.2% and 57.2% after tunning. SVM classifier gives an overall accuracy of 58.6% and 60.3% after tunning. Naïve Bayes classifier gives an overall accuracy of 34.0% which is too low compared to decision tree and SVM classifier. Since both decision tree and SVM classifier have only a small difference between the accuracy before and after tuning, the process maximized the model's performance without overfitting and therefore SVM classifier looks the best.

b) Are there any categories that all classifiers seem to have trouble with?

Since I have calculated the entropy before and after split, I realized that the entropy gained from 1.37 to 1.48 after the entropy split, which means by splitting the “functionary” column, our entropy increases, hence the functionary category has a high level of disorder and it is not suitable to use it as a category for any classifier for training and testing the data set.

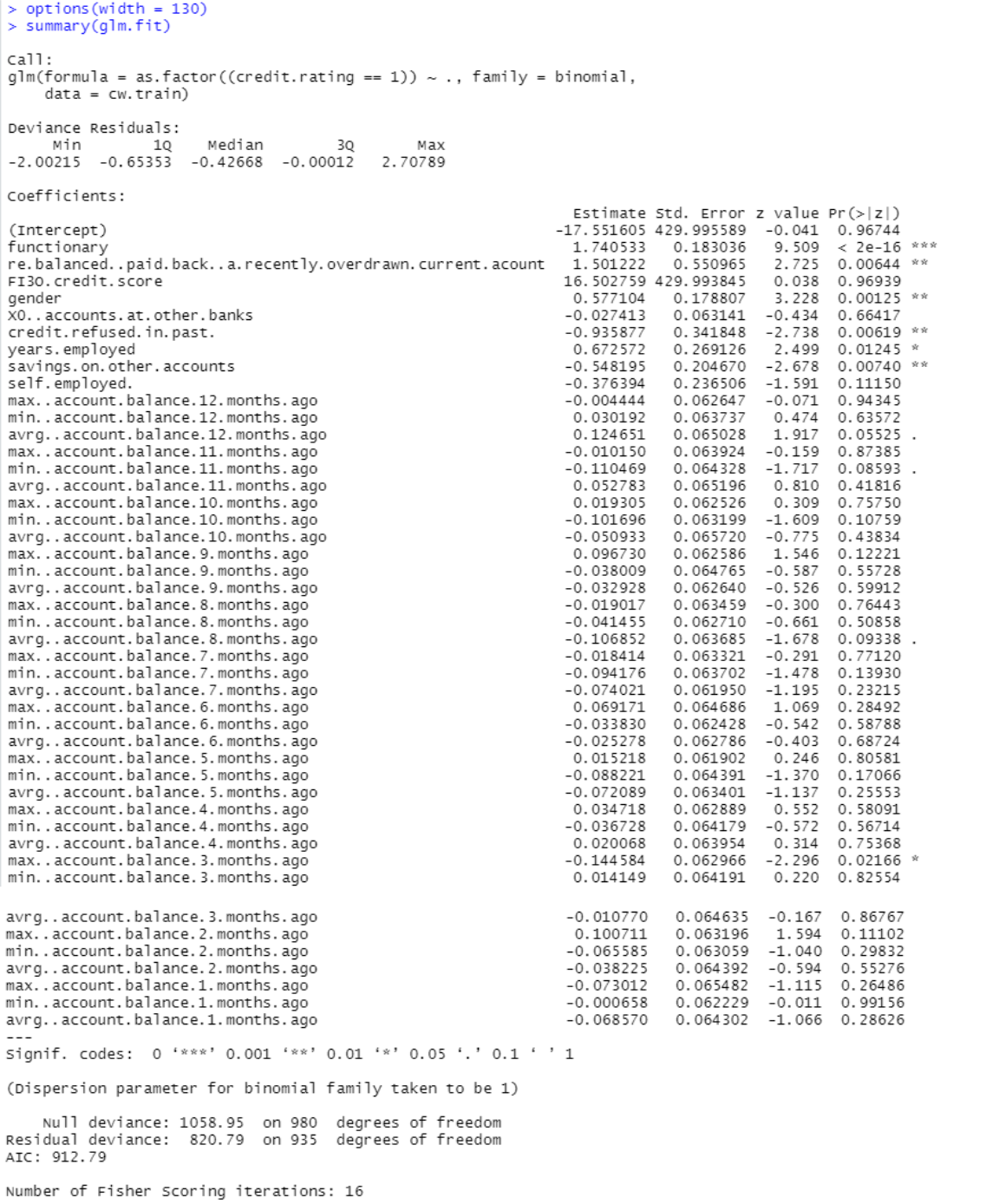
**Question 6:**

a)



b)

Report the summary of the model



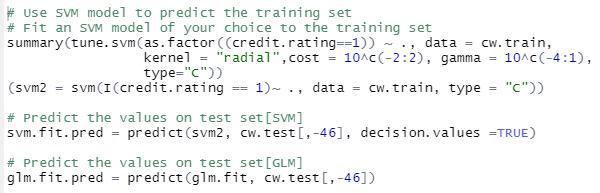
c)

Based on the summary above, we can see “functionary” and

“re.balanced..paid.back..a.recently.overdrawn.current.acount” are appear to be significant since they have the highest positive estimate rate.

However, “FI3O.credit.score” is likely to be spuriously since the estimate rate and std.error is times higher than other predictors.

d)



e)

