Comp 309 Assignment3

Vincent Yu

September 2018

1 Part 1

1.1 Business Objective

Construct a model with the data set giver to predict the total income of different families.

1.2 Data Understanding

In this assignment, a CSV file called census-comp309.csv was provided. The data-set contains 58 features and 14 classes. It contained 7,621 instances. The first 3000 instances in the data-set was used to construct out system. The remaining 4,621 are used to evaluate the system we created. The class label are missing among these 4621 data. In this data-set we have different types features, most of the features are Categorical, and the rest are numerical values. There are missing values in the data-set which need to fix. The data set is unbalanced,the number of low income family is much smaller than the high income family. Features are not independent to each others, such as the race of child related to their parents.

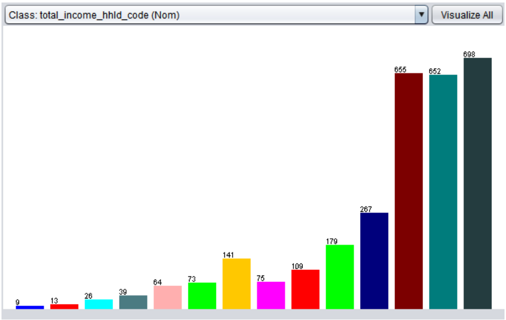
1.3 Data Visualization

1.3.1 Missing values

As mentioned in data understanding, there are lots missing values in the data set. Missing value may cause serious problems, because most classifier eliminate the case with missing values, this could cause we don't have enough data to construct a reliable model.Missing data can also lead to misleading results.

1.3.2 Imbalanced class

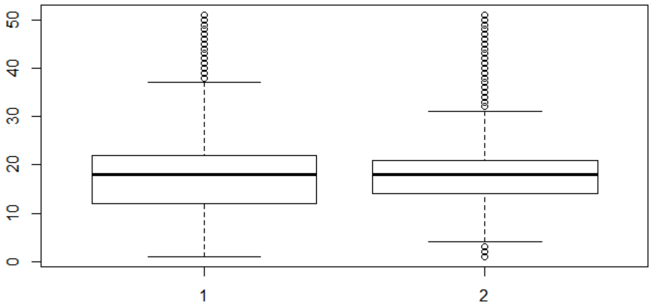
As the figure shown below, it shows the distribution of family in come levels in the training set. The figure illustrates the family with high income dominate the data-set, this may affect the model to more likely predict the family has high income later on.



figure(1)

The model construct by a highly imbalanced dataset is usually not reliable. Because let's say all For this data set, it may predict most family as high income level. Understanding different features is important, because this could help us decided how to preprocessing the data later on.

1.3.3 Outliers

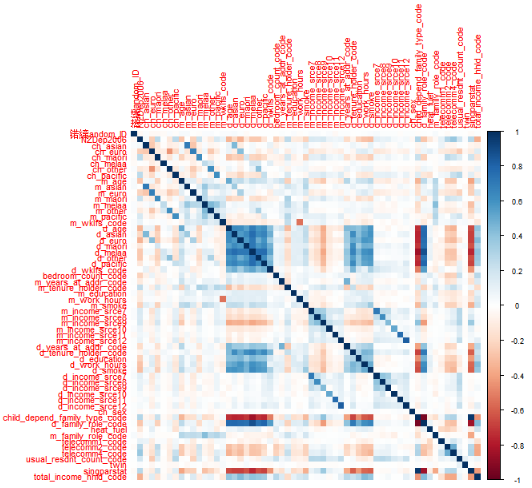


figure(2)

dad-age mom-age

As figure2 shown above, it illustrates the data of the dad's and mom's age in the training set. There are several outliers in these features, the highest age of mother in the dataset is 99. Outlier can ruin the model, thus we need to fix this problem. There are also lots outliers in different features.
To explore more about the data set, we need to find the correlation between different features and the class, the graph below shows the correlation matrix.

1.3.4 Correlation



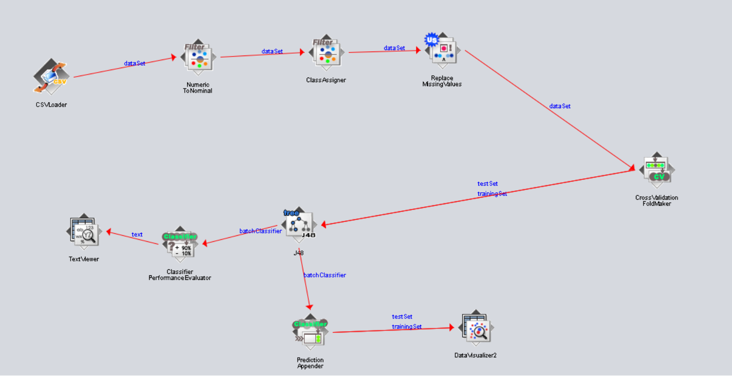
figure(3)

As the figure is shown above, it shows the features highly correlate to the class are the age of father, mother employed full-time, father total hours in work, child depend family type, singparstat, access to cell phone, access to telephone and access to internet . All these features are more likely correlate to the total family income stats. It make sense, the more working hours may point to higher income in real life. And the family has two parents means both mom and dad can support the family, this could make the income higher than single parents family.
Thus we should take more care about the features highly correlate to the class to make a reliable model later on.

2 Part2

2.1 Initial Design

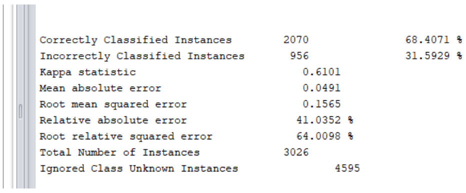
The initial design was done by Weka. The construction is shown below.



figure(4)

The dataset was using is the total data set which include training set and test set. It was imported by using csv loader in pipeline, then it was applied into the numeric to nominal filter to set the class to categorical.

The classifier was using J48 which is one of the techniques from Decision Tree class. This classifier will use the training set to generate a "Tree"as a model which can make predictions later on. But as mentioned before there are lots missing values in the dataset. This made the model not reliable. So there is a missing-value filter applied in the pipeline. The filter will fill the missing value in one attribute with the mean value of that attribute. Then the data was applied into the crossvalidation maker. It was set to 10 folds. The prediction appender applied the model to the test set and made predictions based on the data in the test set. The accuracy of this system is showing below.



```
Correctly Classified Instances        2070               68.4071 %
Incorrectly Classified Instances       956               31.5929 %
Kappa statistic                          0.6101
Mean absolute error                      0.0491
Root mean squared error                  0.1565
Relative absolute error                 41.0352 %
Root relative squared error             64.0098 %
Total Number of Instances             3026
Ignored Class Unknown Instances                 4595
```

figure(5)

As figure 5 is showing above, obviously this model is not good. One reason of this system was not reliable is the way to impute missing values. For this data set, most attributes are categorical, the filter replace the missing value with mean of that attribute. Let's take the dad-euro attribute as an example. This attribute has two levels (0 and 1). The mean value of this data set is around 0.3. In Weka it will treat this attributes as an numeric attributes instead of a categorical data. This cam be fixed by round the data to an integer and set this attributes to nominal. The re-sample filter was also included in this system, but it change the initial data set, which affected the number of instances. After interrupting it is very hard to make the number of instances match the initial number.

2.2 intermediary systems

2.2.1 Second System

As mentioned in part 1, there are outliers in different features. J48 is not good at handling outliers. So I did clustering in these features. Take m-age as an example, most data are in the range from 25 to 40, but the greatest age is 96. I fixed the data set by capping the data, and assigned the outliers with the function usually used in statistic science. The formula of handling outliers is showing below:

$$IQR = Q3 - Q1(Q1 : lowerquartile, Q3 : upperquartile) \tag{1}$$

$$if(data > Q3)thenreplacethedatawithQ3 + 1.5 \times IQR \tag{2}$$

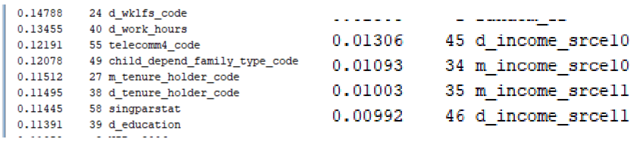
$$else(data < Q1)thenreplacethedatawithQ1 - 1.5 \times IQR \tag{3}$$

This process were applied to all the numeric attributes. The accuracy improved little bit(26.9%) in Weka, but the accuracy on Kaggle decrease little bit(24.95%). I think it may caused by overfitting. J48 (ID3) can overfit to the training data. I did research online find out J48 is harder to handle on continuous data. With continuous data, the tree construct by J48 can be very large. But to avoid overfitting, smaller decision trees should be preferred.

2.2.2 Third System

As mentioned before, J48 is not the best suitable classifier for our dataset. So I changed it to Random Forest. It is an ensemble learning method for classification, the operate by constructing a multitude of decision tress using training. Random forest correct for decision trees'habit of overfitting to their training set.It can also automatically ignore the instances with unknown class label when we constructing the model. So it not necessary to split the dataset into training set and testing set, we can just use cross validation.

There was a huge increment in accuracy by using Random Forest. I used the raw data without any modification, the accuracy became around 30%. Imported the data after fixing outliers, the accuracy increased about 1%. Then I run the attributes correlation in Weka.



```
0.14788    24 d_wklfs_code
0.13455    40 d_work_hours
0.12191    55 telecomm4_code
0.12078    49 child_depend_family_type_code
0.11512    27 m_tenure_holder_code
0.11495    38 d_tenure_holder_code
0.11445    58 singparstat
0.11391    39 d_education
```

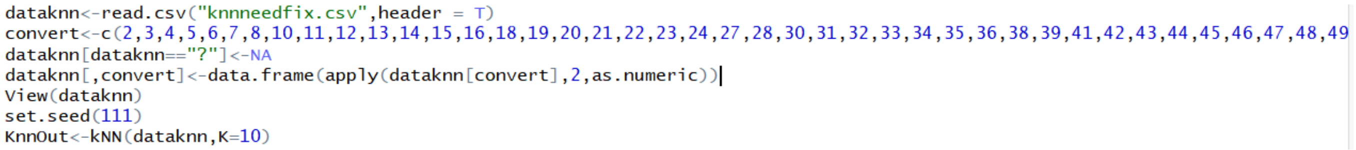
```
0.01306    45 d_income_srce10
0.01093    34 m_income_srce10
0.01003    35 m_income_srce11
0.00992    46 d_income_srce11
```

figure(6)

As the graph is showing above, the left graph indicate the most correlate attributes. The result of correlation obtained now is consistent with the previously obtained correlation matrix. The most correlate the attributes are dad's work type, working hours, and child depend family type. This makes sense, for parents have full time job normally can earn more money and the more working hours normally means higher income. The graph on the right side of figure 6 shows the less correlate attributes. Removed the least correlate attribute, the accuracy increased 0.5%.

2.2.3 Forth System

In this system, I changed the way for imputing missing value in R.



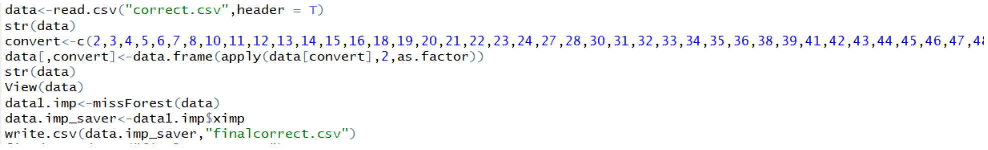
```r
dataknn<-read.csv("knnneedfix.csv",header = T)
convert<-c(2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19,20,21,22,23,24,27,28,30,31,32,33,34,35,36,38,39,41,42,43,44,45,46,47,48,49
dataknn[dataknn=="?"]<-NA
dataknn[,convert]<-data.frame(apply(dataknn[convert],2,as.numeric))
View(dataknn)
set.seed(111)
KnnOut<-kNN(dataknn,K=10)
```

The first line read the csv file, and the second line set the right type of each features.(Note: after reading the csv file into R it automatically recognize most the features as factor, but KNN works better for numeric values, so here convert all the features into numeric.) Here k was set to 10. The value of K is highly depend on the data. In general a larger k suppresses the effects of noise. So k was set to a relatively high value.

After fixing missing values, I tried to run it in Weka with using Random forest as my classifier, the accuracy was higher than using mean value replace the missing values. And the accuracy on Kaggle I got 0.30800.

2.2.4 Final System

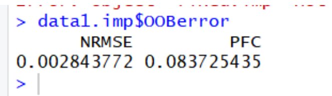
The missing value imputation of this system was done by using Miss Forest package in R. MissForest is a non-parametric imputation method for basically any kind of data. It can deal with mixed-type of variables, nonlinear relations, complex interactions and high dimensionality. This suits the dataset due to our data set mixed with numeric and categorical types.



```
data<-read.csv("correct.csv",header = T)
str(data)
convert<-c(2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19,20,21,22,23,24,27,28,30,31,32,33,34,35,36,38,39,41,42,43,44,45,46,47,4
data[,convert]<-data.frame(apply(data[convert],2,as.factor))
str(data)
View(data)
data1.imp<-missForest(data)
data.imp_saver<-data1.imp$ximp
write.csv(data.imp_saver,"finalcorrect.csv")
```

figure(7)

The code showing above shows how miss forest impute the missing values in this system.

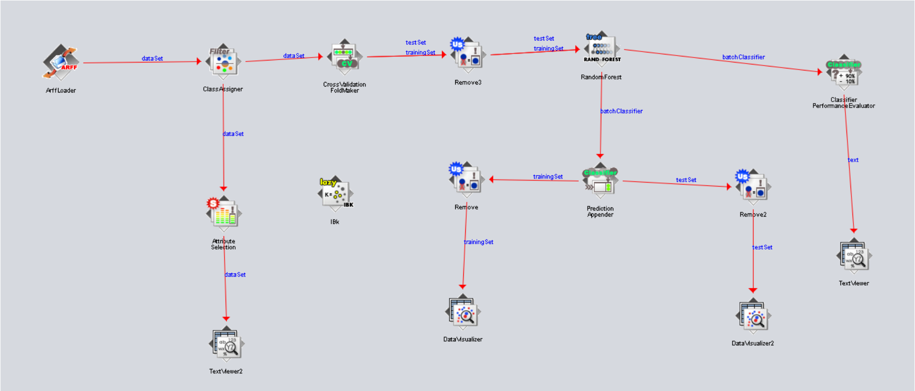


```
> data1.imp$OOBerror
      NRMSE         PFC 
0.002843772 0.083725435 
> 
```

figure(8)

Figure 8 shows the Out of Bag error, it measures the the prediction error of miss forest. As mentioned before, the data set mixed numeric and categorical data, so the OOB supplies two values for the result of the imputation. The first value is the normalized root mean squared error(0.00284..). The second value is the proportion of falsely classified entries(PFC) in the categorical part of the imputed data set.(0.0837..). From this two values, it confirms the imputation for missing value was acceptable. I did not apply the process for outliers in this system, because RF is not sensitive to outliers.

After preprocessing the data in R, I saved it as CSV file. Before importing this data into Pipeline, I manually changed the file from CSV to ARFF, and manually set the type of attributes. Because importing data from CSV into weka, it will automatically assign the type for all the attributes. I manually assigned the right type to all the attributes.



figure(9)

The remove filter remove the least correlated attribute.
0.00992 45 d-income-srce10

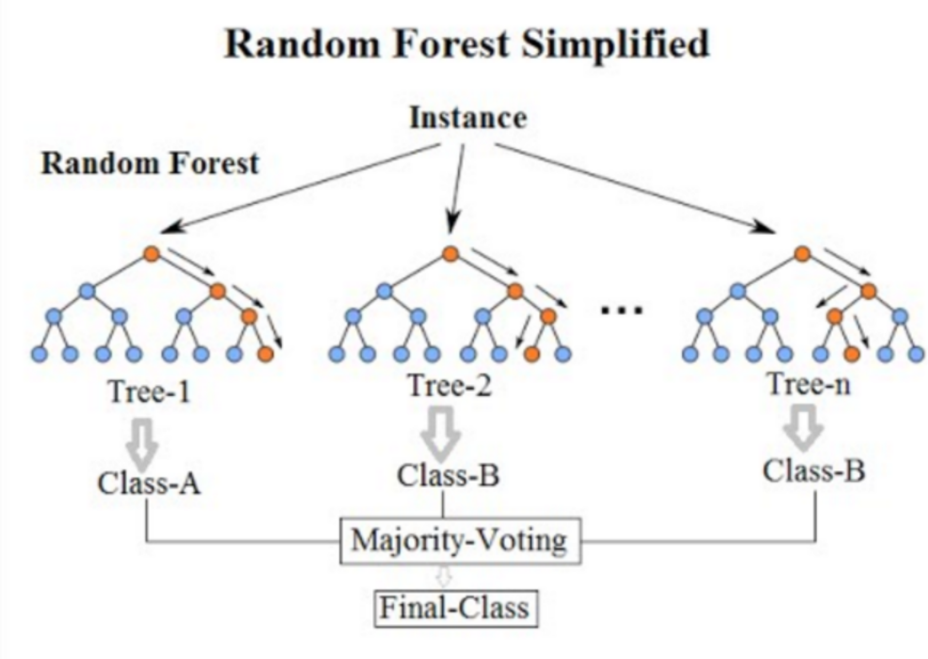
I set number of features to 7, which selects 7 attributes each time to compute one tree. And I set the maximum depth of the tree to 10 to avoid overfitting. The last thing I changed was the number of iteration. I changed it from 100 to 150 to make the model more accurate.

The accuracy of this system on Kaggle is 0.349.

But I think for this data set, we can also do clustering, which combining the similar attributes. This operation can reduce the number of attributes. As mentioned before, less attributes can leads to smaller "*Tree*", it can avoid overfitting. It can help to build a model more reliable. Clustering can based on the correlation between different features, we can combine the attributes which are highly correlated to each others.

3 Part3

The final design is not easy to interpret. As mentioned before, random forest randomly choose features to compute one tree, and it will compute several trees in one iteration.(The number of trees depend on the number of features and the number of selecting features to compute one tree) After all the iterations it will vote a best model based on the accuracy. It will take lots time to evaluate the model built by Random Forest.



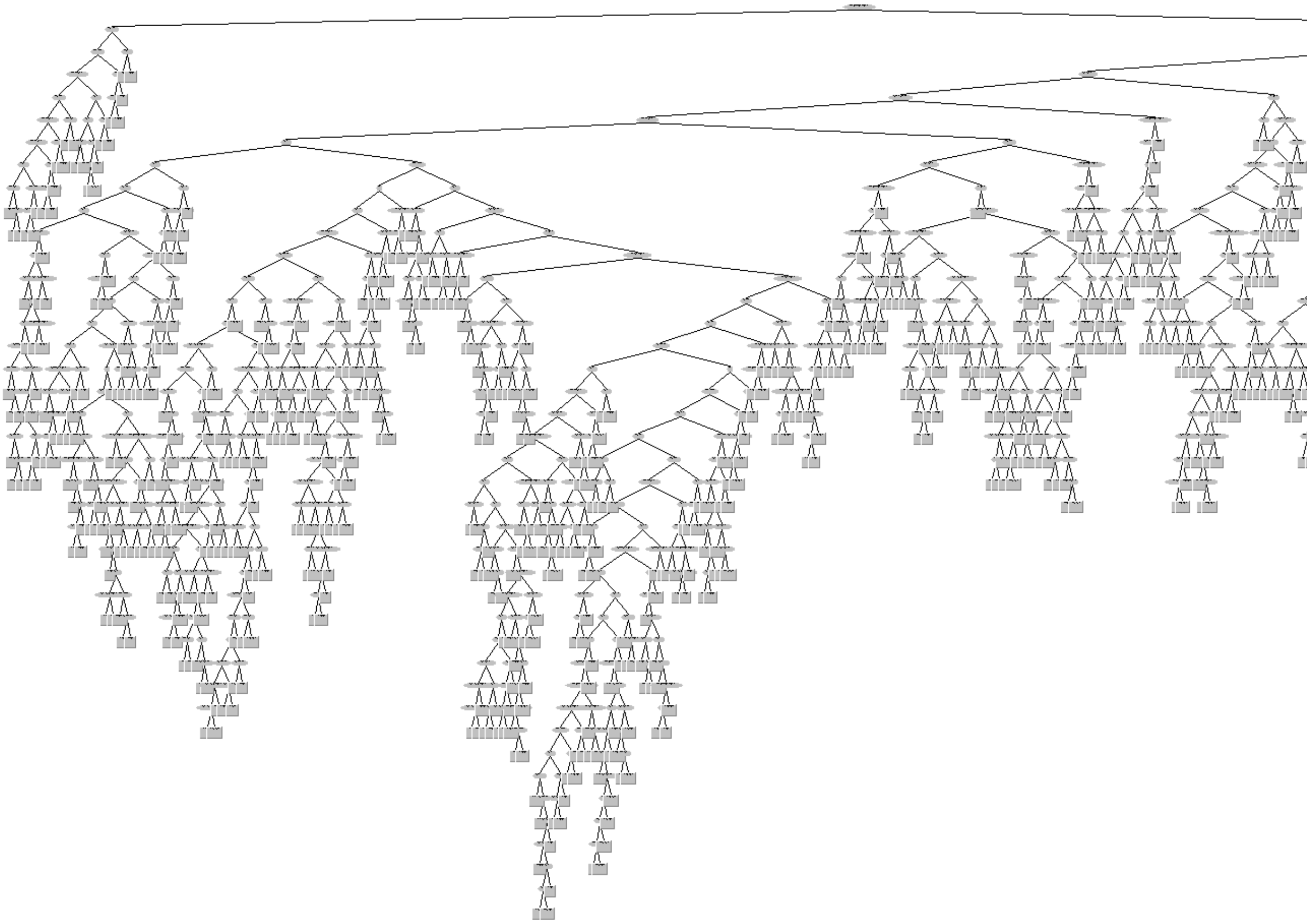
figure(10)

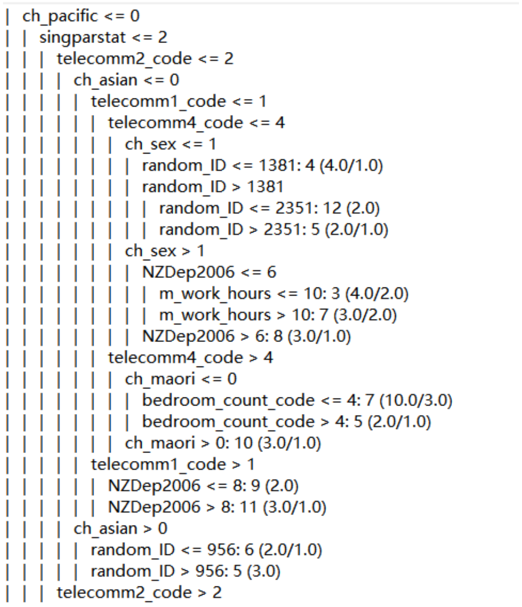
The ethical consequence of using random forest is that, it is more likely to predict the income of European family as higher than that of Asian family.
The initial system is much easier to interpret, it was made by using J48. The accuracy of this system is 0.249. Run this system using explore in Weka, it can generate a tree represent this system. Although the tree generate by j48 is much bigger than a single tree generate by Random Forest. But we still can easily visualize how the tree is pruned. Decision tree is a good technique to predict categorical classes. But for our data, it is not the best technique. Because we have too many features which will lead to a very big decision tree. Also the data for training is highly imbalanced, this leads to the tree is biased. J48 is easier to interpret but the model is vulnerable, it easy overfitting, and the performance for our data is much lower than Random forest, but on the other hand, the complexity of J48 is much lower than Random forest. The time to construct a system using J48 is much less than using Random forest. This is a trade off. So for our assignment, if we want to have a visualization of the tree, I recommend using J48 but if we want to gain a high accuracy I think Random Forest is the most suitable classifier.

If we are asking to produce a system that was not biased towards certain population groups, I will re-sample the data, to make the data set become balanced. And do clustering among attributes of different ethnic groups.

4 Appendix

4.0.1 Tree generate by J48





```
| ch_pacific <= 0
| | singparstat <= 2
| | | telecomm2_code <= 2
| | | | ch_asian <= 0
| | | | | telecomm1_code <= 1
| | | | | | telecomm4_code <= 4
| | | | | | | ch_sex <= 1
| | | | | | | | random_ID <= 1381: 4 (4.0/1.0)
| | | | | | | | random_ID > 1381
| | | | | | | | | random_ID <= 2351: 12 (2.0)
| | | | | | | | | random_ID > 2351: 5 (2.0/1.0)
| | | | | | | ch_sex > 1
| | | | | | | | NZDep2006 <= 6
| | | | | | | | | m_work_hours <= 10: 3 (4.0/2.0)
| | | | | | | | | m_work_hours > 10: 7 (3.0/2.0)
| | | | | | | | NZDep2006 > 6: 8 (3.0/1.0)
| | | | | | telecomm4_code > 4
| | | | | | | ch_maori <= 0
| | | | | | | | bedroom_count_code <= 4: 7 (10.0/3.0)
| | | | | | | | bedroom_count_code > 4: 5 (2.0/1.0)
| | | | | | | ch_maori > 0: 10 (3.0/1.0)
| | | | | telecomm1_code > 1
| | | | | | NZDep2006 <= 8: 9 (2.0)
| | | | | | NZDep2006 > 8: 11 (3.0/1.0)
| | | | ch_asian > 0
| | | | | random_ID <= 956: 6 (2.0/1.0)
| | | | | random_ID > 956: 5 (3.0)
| | | telecomm2_code > 2
```

4.0.2 code

```
library(foreach)
library(iterators)
library(itertools)
library(ggplot2)
library(gridExtra)
library(corrplot)
library(randomForest)
library(ggplot2)
library(gridExtra)
library(cowplot)
library(missForest)
library(dplyr)
data<-read.csv("correct.csv",header = T)
str(data)
convert<-c(2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19,20,21,22,23,24,27,28,30,31,32,33,3
data[,convert]<-data.frame(apply(data[convert],2,as.factor))
str(data)
View(data)
data1.imp<-missForest(data)
data.imp_saver<-data1.imp$ximp
write.csv(data.imp_saver,"finalcorrect.csv")
data1.imp$OOBerror
fixed<-read.csv("finalcorrect.csv")

train<-read.csv("lalalala.csv",header = T)
View(train)
```

```
names(train)[1]<-"ID"
View(train)
test<-read.csv("testfinal.csv",header = T)
test[test=="?"]<-NA
names(test)[1]<-"ID"
View(test)
convert<-c(2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19,20,21,22,23,24,27,28,30,31,32,33,3
train[,convert]<-data.frame(apply(train[convert],2,as.factor))
str(train)
convert<-c(2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19,20,21,22,23,24,27,28,30,31,32,33,3
test[,convert]<-data.frame(apply(test[convert],2,as.factor))
set.seed(111)
rf <-randomForest(total_income_hhld_code~.,data=train)
print(rf)
attributes(rf)
library(caret)
p1 <-predict(rf,train)
head(p1)
library(e1071)
confusionMatrix(p1,train$total_income_hhld_code)
f<-factor(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14),exclude = NULL)
length(levels(f))
levels(f)
train1<-train
train1$total_income_hhld_code<-factor(as.character(c(1:14,NA)),exclude = NULL)
str(train1)
test1<-test
View(test)
str(test1)
#test1$total_income_hhld_code<-factor(c(1:14,NA),exclude = NULL)
p2<-predict(rf,test)
levels(test1$total_income_hhld_code)<-c(levels(test1$total_income_hhld_code),"1","2","
library(VIM)
dataknn<-read.csv("knnneedfix.csv")


str(knnneedfix1)
fixing<-knnneedfix1
str(fixing)

library(Hmisc)
library(DMwR)
library(VIM)
da00<-read.csv("knnneedfix.csv",header=T)
da00[da00=="?"]<-NA

View(da00)
KnnOut<-knnImputation(da00,k=10,scal=T,meth="weighAvg" )
```

```
impute()
View(KnnOut)
write.csv(knnOut,"knnfixed.csv")
```