UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Analysis and Model Selection on the adult dataset

MACHINE LEARNING COURSE PRACTICE

Han Yang

June 2016

1. Introduction

We decide to analysis the Adult Dataset [1] in UCI machine learning repository, extracted from the census bureau database of the US GOV. The dataset deals with a binary classification problem, that is to determine whether a person makes over 50K a year.

It has 32561 individuals with 14 attributes, 6 are continuous attributes and 8 are categorical attributes. Here is the detailed information:

Variables		
Age	Continuous	The age of the individual
Workclass	Categorical	The type of employer the individual has
Fnlwgt	Continuous	The # of people the census takers believe that observation
Native_country	Categorical	Country of origin for person
Capital_gain	Continuous	Capital gains recorded
Capital_loss	Continuous	Capital Losses recorded
Relationship	Categorical	Contains family relationship values.
Race	Categorical	Descriptions of the individual's race.
Sex	Categorical	Biological Sex
Hrs_per_week	Continuous	Hours worked per week
Marital_status	Categorical	Marital status of the individual
Occupation	Categorical	The occupation of the individual
Education_num	Continuous	Highest level of education in numerical form
Education	Categorical	Highest level of education achieved for that individual
Income_class	Categorical	Whether or not the person makes more than $50,000

The dataset was first cited in Scaling Up the Accuracy of Naive-Bayes Classifiers [2], and then been used in many papers. So we think it's a classic machine learning dataset for which we can do a deep exploration inside.

In this project, I will do a deep data preprocessing on the dataset, use the cooked dataset to fit several classifiers, pick up the best model and analysis its generalization error.

2. Related previous works

The adult dataset is a classic dataset in UCI Machine Learning Repository. Several methods [3], [4] have been implemented on this dataset, their result is as followed:

	Algorithm	Error
1	C4.5	15.54
2	C4.5-auto	14.46
3	C4.5 rules	14.94
4	Voted ID3 (0.6)	15.64
5	Voted ID3 (0.8)	16.47
6	T2	16.84
7	1R	19.54
8	NBTree	14.10
9	CN2	16.00
10	HOODG	14.82

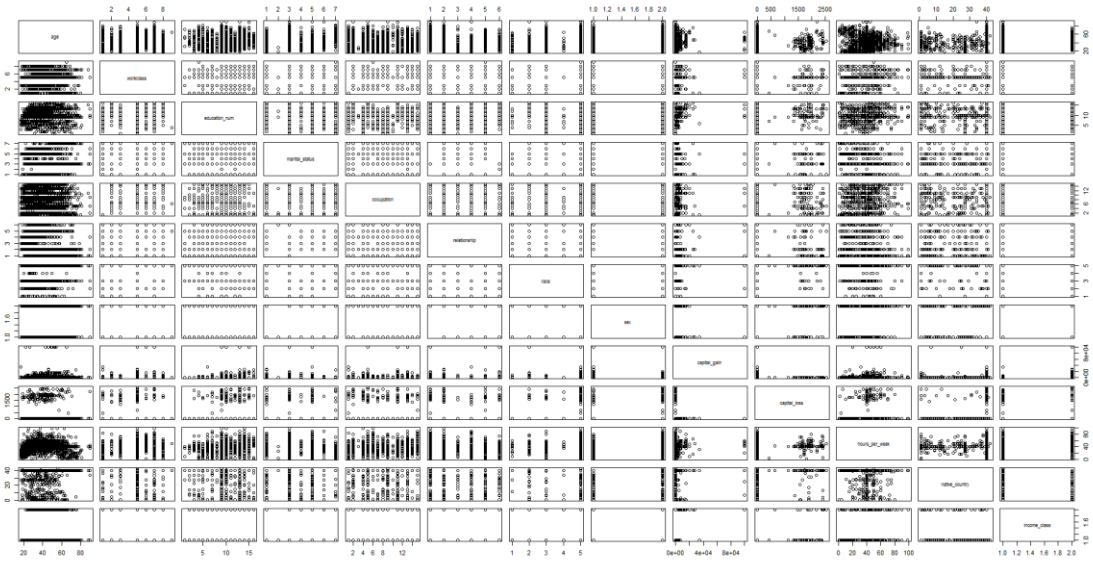
The best result is among 14% test error. But one thing to take consideration is that these results are based on the raw data without any data cleaning. As it is known, the quality of the data has a great impact on the quality of the model, as “Garbage in garbage out”. In this project, I’ll try several data preprocessing technique, and try to get a better result based on that.

3. Data exploration process

3.1. Pre-processing

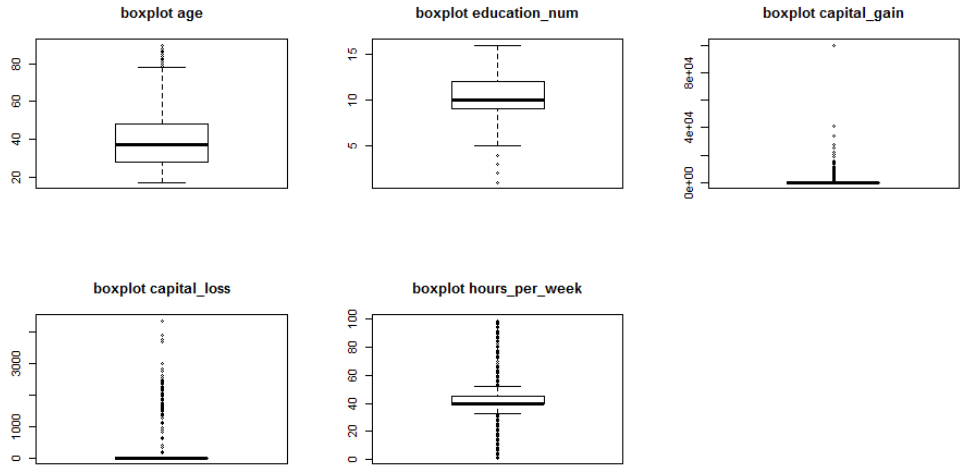
First of all, I will delete the variables “fnlwgt”, “education”. Fnlwgt is the sampling weight, I don’t need it for building my model. And education is the same as education_num, I decide to use the continuous variable education_num, so I will delete education from the dataset.

Next, we take a look on the distribution of every attribute.



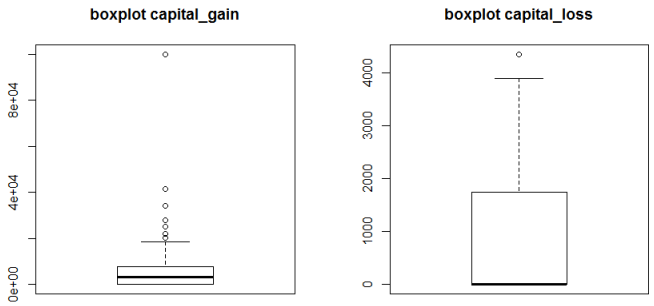
From the plot, we can see that some variables are equally distributed for all its values, but some are not equally distributed for all its values. The no-equal situation is not good for building the machine learning model, so we need to do some data cleaning.

3.1.1. Pre-processing - Continuous Variables



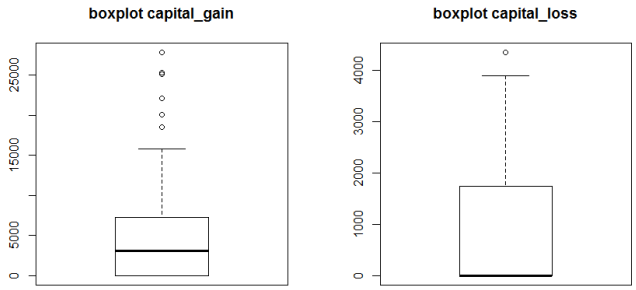
From the boxplot of the five continuous variables, we can clearly see that captital_gain and capital_loss have most of their values with zero, which is not good for us to build the model. So I decide to delete all the values with 0 in both capital_gain and capital_loss. (This means I will delete the individual only if 0 is in both variables)

```
> X.adult.caploss= X.adult[X.adult$capital_loss!=0,]
> X.adult.capgain= X.adult[X.adult$capital_gain!=0,]
> X.adult = rbind(X.adult.caploss, X.adult.capgain)
```



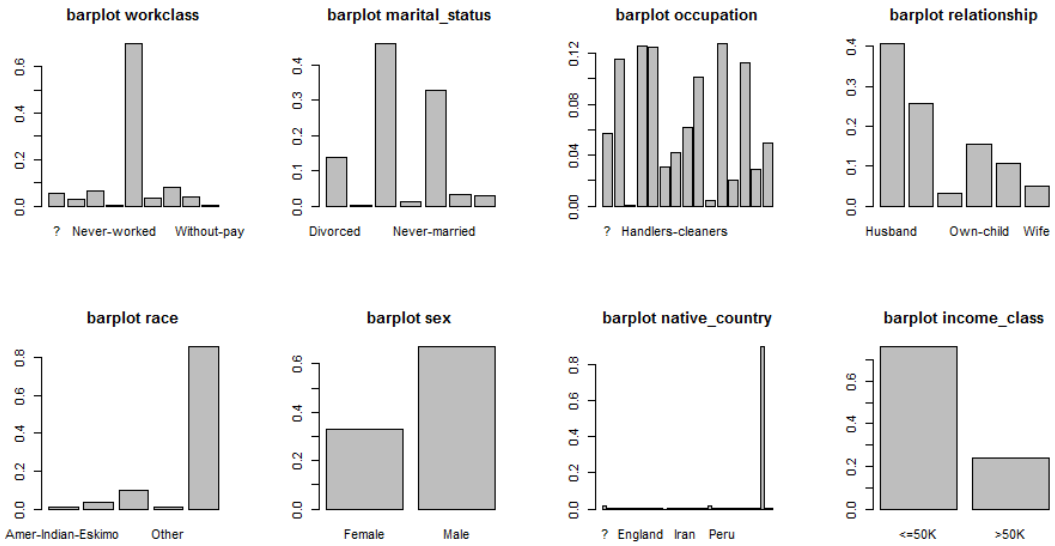
From the plot, we can see that we get a good capital_loss variable, but the capital_gain is not so good because it contains some outliers. Then I calculate the The interquartile range (IQR), and delete the values bigger than Q3+3IQR, which will be our outliers.

```
> max <- quantile(X.adult$capital_gain,0.75, na.rm=TRUE) + (IQR(X.adult$capital_gain,
na.rm=TRUE) * 3)
> X.adult <- X.adult[X.adult$capital_gain < as.numeric(max),]
```



This time we have a really good distribution of capital_gain and capital_loss.

3.1.2. Pre-processing - Categorical Variables



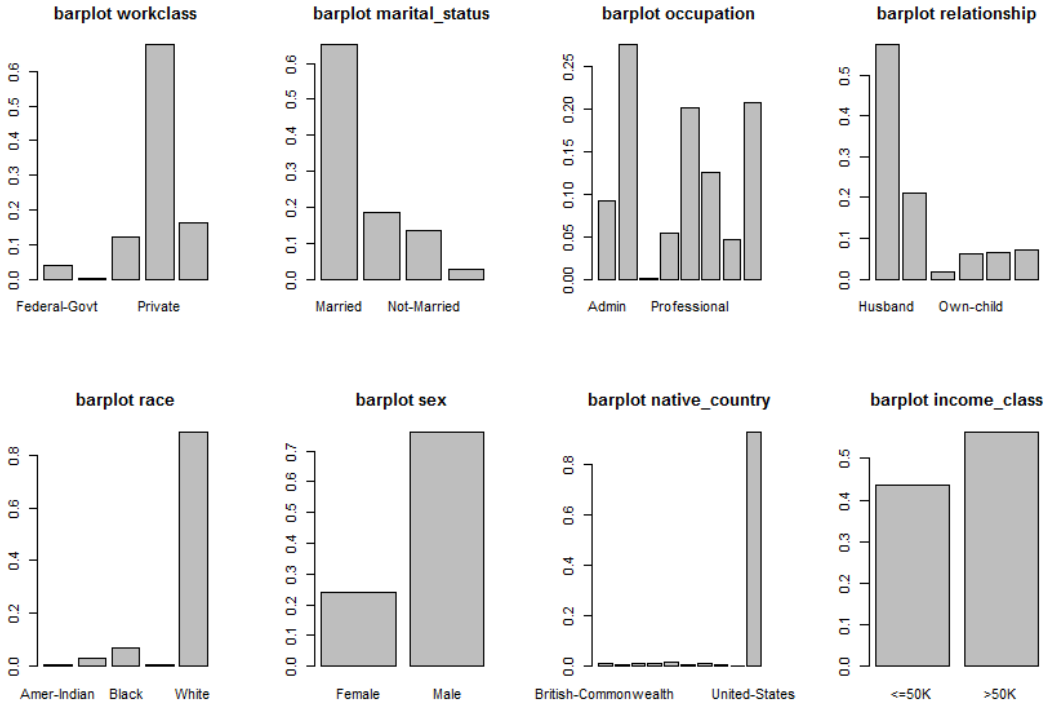
From the barplot of the categorical variables, we can see there are some categorical variables with lots values for one value and few values for some other values. In this case, we can combine some values into one, thus they'll be more equally distributed. [5]

The combine (block) information is in the following table:

Variable	Vaule	Value after combination
marital_status	Married-AF-spouse	Married
	Married-civ-spouse	
	Married-spouse-absent	Not-Married
	Separated	
	Divorced	
workclass	Local-gov	Other-Govt
	State-gov	
	Without-pay	Not-Working
	Never-worked	

occupation	Farming-fishing	Blue-Collar
	Handlers-cleaners	
	Machine-op-inspct	
	Transport-moving	
	Priv-house-serv	Service
	Other-service	
native_country	countries within the Eurozone that more affluent	Euro_1
	countries within the Eurozone that less affluent	Euro_2
	formerly British holdings	British-Commonwealth

After the blocking, we plot again the categorical data to see the changes.



We can see that the categorical variables are more equally distributed now, which will helps to improve the accuracy of prediction of our model.

3.1.3. Pre-processing - Missing Values

There are some NA values in the dataset as followed:

```
> summary(X.adult[,c(2,5,12)])
         workclass          occupation                native_country
 Federal-Govt : 151   Blue-Collar :1070   United-States       :3702
 Not-Working  :   2   White-Collar: 804   Latin-America       :  56
 Other-Govt   : 478   Professional: 783   British-Commonwealth:  51
 Private      :2629   Sales       : 487   SE-Asia             :  39
 Self-Employed: 630   Admin       : 356   Euro_2              :  34
 NA's         : 175   (Other)     : 390   (Other)             : 100
                      NA's        : 175   NA's                :  83
```

Here I use the Multivariate Imputation by Chained Equations (MICE) method to impute the missing values.

```
library(mice)
X.adult.mice <- mice(X.adult, MaxNWts = 10000)
X.adult.complete = complete(X.adult.mice)
```

What mice() does is that it impute all of the missing values by doing KNN for several times.

Now I finish with the data-preprocessing part, and get the dataset as follows:

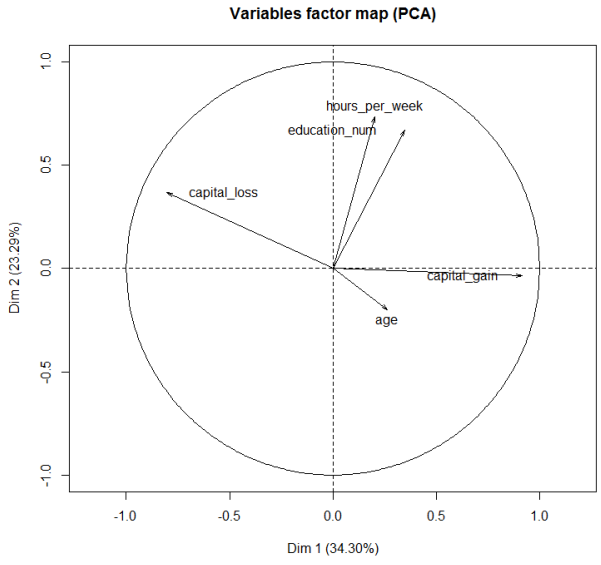
```
> summary(X.adult.complete)
      age              workclass    education_num        marital_status             occupation          relationship
 Min.   :17.00   Federal-Govt : 162   Min.   : 1.00   Married      :2645   Blue-Collar      :1115   Husband       :2332
 1st Qu.:34.00   Not-Working  :   2   1st Qu.: 9.00   Never-Married: 753   White-Collar     : 828   Not-in-family : 856
 Median :42.00   Other-Govt   : 496   Median :10.00   Not-Married  : 554   Professional     : 807   Other-relative:  69
 Mean   :43.08   Private      :2750   Mean   :10.96   Widowed      : 113   Sales            : 511   Own-child     : 253
 3rd Qu.:51.00   Self-Employed: 655   3rd Qu.:13.00                       Admin            : 387   Unmarried     : 270
 Max.   :90.00                        Max.   :16.00                       Other-Occupations: 215   Wife          : 285
                                                                          (Other)          : 202

         race          sex        capital_gain    capital_loss    hours_per_week              native_country income_class
 Amer-Indian:  31   Female: 968   Min.   :    0   Min.   :   0.0   Min.   : 1.00   United-States      :3766   <=50K:1774
 Asian      : 127   Male  :3097   1st Qu.:    0   1st Qu.:   0.0   1st Qu.:40.00   British-Commonwealth:  57   >50K :2291
 Black      : 277                 Median : 3103   Median :   0.0   Median :40.00   Latin-America      :  57
 Other      :  21                 Mean   : 4658   Mean   : 699.3   Mean   :43.19   SE-Asia            :  43
 White      :3609                 3rd Qu.: 7298   3rd Qu.:1741.0   3rd Qu.:50.00   Euro_2             :  39
                                  Max.   :27828   Max.   :4356.0   Max.   :99.00   Euro_1             :  32
                                                                                   (Other)            :  71
```

3.2. Visualization

I use the PCA and MCA in the factoMineR package to do the visualization.

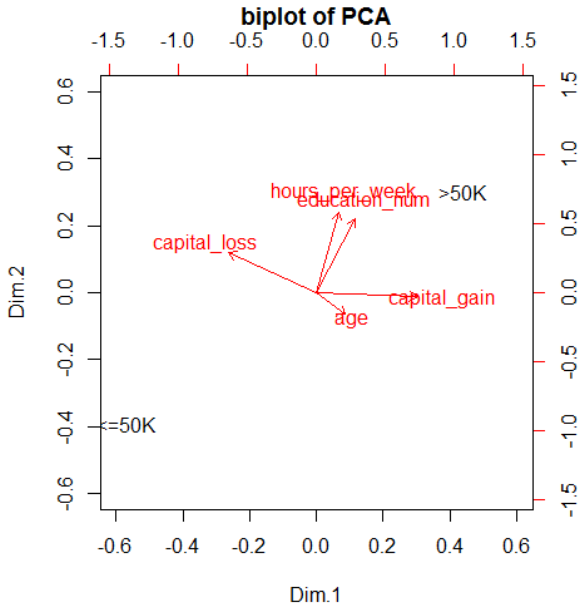
```
library(FactoMineR)
PCA = PCA(X = X.adult.complete, quali.sup = c(2,4,5,6,7,8,12,13))
biplot(PCA$quali.sup$coord[c("<=50K", ">50K"),],PCA$var$coord, xlim = c(-0.6,0.6), ylim = c(-0.6,0.6), main = "biplot of PCA")
abline(h=0,v=0,col="gray")
MCA = MCA(X = X.adult.complete[,c(2,4,5,6,7,8,12,13)], quali.sup = 8)
```



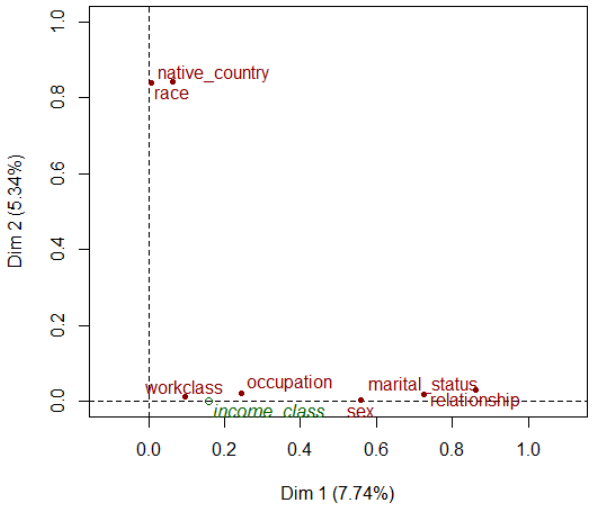
From the PCA plot, we can see that the “education_num” and the “hours_per_week” have strong correlation. It means that the more educated a person is, the more time he will spend on his working.

Another thing is the opposite correlation of “capital_loss” with “capital_gain” and “age”. This means that the elder a person is, the more capital gain he will receive.

But there is no information shown on the plot about their relationship with income_class. So I do a biplot() of the continuous variables and the income_class.



From the biplot, we can see that both dimensions of the factors will effect on the income_class. With more “hours_per_week”, “education_num”, “capital_gain” you will get more income, and vice versa.



From the plot of MCA about the representation of active categories on the first plane, we can see that the income_class has a R^2 of 0.2 in the first dimension, and a R^2 of 0 in the

second dimension. Which means only the first dimension has the effect on income_class, while the second dimension counts for no effect on the income_class.

And the variables that mostly affect the first dimension are “relationship”, “marital_status”, “sex”, “occupation”. In another word, these variables are the most significant variables that will effect on the classification of the income_class.

4. Validation protocol

The validation protocol is divided into two parts.

First I separate the whole data set into training data and testing data.

Then I select several classifiers, and use the heuristic method to tune the parameters of those chosen classifier to get the best prediction accuracy of each model.

Finally I choose the best model based with the minimum prediction error. And calculate its generalization error by doing a 10-fold cross-validation and calculating its confidence interval.

The following code is used to split our dataset into the training dataset and test dataset. We use the random sampling method in R to do this step. We sample ¾ of the whole dataset as training dataset, and the rest ¼ as the testing dataset.

```
> N = nrow(X.adult.complete)
> split = 1:N
> size = round(N*3/4)
> split.training = sample(x = split, size = size, replace = FALSE)
> split.testing = split[-split.training]
> X.adult.training = X.adult.complete[split.training,]
> X.adult.testing = X.adult.complete[split.testing, ]
```

Dataset information:

Dataset	4065 individuals
Training data (3/4)	3049 individuals
Testing data (1/4)	1016 individuals

5. Model selection

In this part, I will fit four different classifiers: Naïve Bayes, SVM, Random Forest and Neutral Network. I will calculate their test errors using the testing data to do the prediction. And then I choose the best one to analysis its generalization error.

5.1. Naïve Bayes

Naïve Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

```
> model.naivebayes <- naiveBayes(income_class ~ ., data = X.adult.training)
```

The result table of the testing dataset:

Naïve Bayes	Predicted <=50K	Predicted >=50K
Real <=50K	357	99
Real >=50K	54	506
		Test error: 15.05906%

5.2. Support Vector Machine

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.

In our case, I tried 4 different kernels to build the SVM.

```
> model.svm.linear <- svm(income_class ~ ., data = X.adult.training, type="C-classification", cost=1, kernel="linear", scale = FALSE)
> model.svm.linear <- svm(income_class ~ ., data = X.adult.training, type="C-classification", cost=1, kernel="polynomial", degree=2, coef0=1, scale = FALSE)
> model.svm.linear <- svm(income_class ~ ., data = X.adult.training, type="C-classification", cost=1, kernel="polynomial", degree=3, coef0=1, scale = FALSE)
> model.svm.linear <- svm(income_class ~ ., data = X.adult.training, type="C-classification", cost=1, kernel="radial", scale = FALSE)
```

The result table of the testing dataset:

SVM linear kernel	Predicted <=50K	Predicted >=50K
Real <=50K	413	43
Real >=50K	396	164
		Test error: 43.20866%
SVM quadratic kernel	Predicted <=50K	Predicted >=50K
Real <=50K	349	107
Real >=50K	213	347
		Test error: 31.49606%

SVM cubic kernel	Predicted <=50K	Predicted >=50K
Real <=50K	296	160
Real >=50K	217	343
		Test error: 37.1063%
SVM RBF Gaussian Kernel	Predicted <=50K	Predicted >=50K
Real <=50K	444	12
Real >=50K	54	506
		Test error: 6.496063%

As we can see, the SVM with RBF Gaussian kernel has the best result, with a test error around 6.5%.

5.3. Random Forest

Random forest uses a number of decision trees, in order to improve the classification accuracy rate.

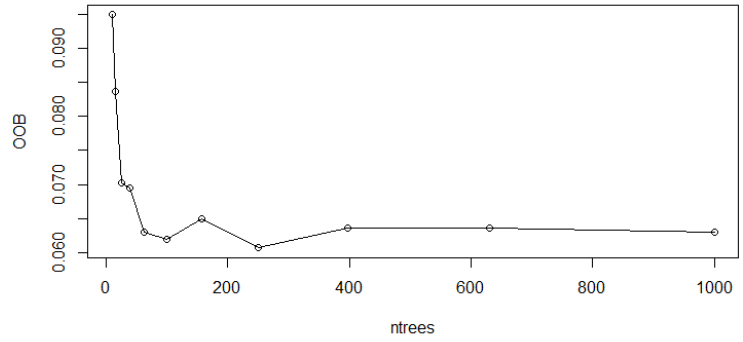
To implement a random forest in R, we need to specify two parameters, one is the mtry value: Number of variables randomly sampled as candidates at each split. Another one is the ntree value: Number of trees to grow.

As we are doing a classification work, we choose the mtry value as sqrt(p), p is the number of explanatory variables. This mtry is the default value.

And we do a heuristic method for choosing the ntree variable. We try different ntree values from 1 to 1000, with the split of 10^seq(1,3,by=0.2).

```
library(randomForest)
(ntrees <- round(10^seq(1,3,by=0.2)))
rf.results <- matrix (rep(0,2*length(ntrees)),nrow=length(ntrees))
colnames (rf.results) <- c("ntrees", "OOB")
rf.results[,"ntrees"] <- ntrees
rf.results[,"OOB"] <- 0
ii <- 1
for (nt in ntrees)
{
   print(nt)
   model.X.random <- randomForest(income_class ~ ., X.adult.training, ntree=nt, proximity=FALSE)
   # get the OOB
   rf.results[ii,"OOB"] <- model.X.random$err.rate[nt,1]
   ii <- ii+1
}
```

The result is:



As we can see, the OOB error of the random forest reach its minimum when the ntree is around 300, so we choose 300 as the ntree.

```
model.X.random <- randomForest(income_class ~ ., X.adult.training, ntree=300, proximity=FALSE)
model.X.random.pred = predict(model.X.random, X.adult.testing, type = "class")
```

The result table is:

Radom Forest	Predicted <=50K	Predicted >=50K
Real <=50K	407	13
Real >=50K	39	557
		Test error: 5.111811%

5.4. Neural Networks

An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain.

Here we build a single hidden layer neural network using nnet(). We start with 541 weights and regular the model during the process.

```
> library(nnet)
> model.nnet <- nnet(income_class ~., data = X.adult.training, size=10, maxit=20000,
decay=0.01)
# weights:  541
initial  value 2809.523119
iter  10 value 2012.164540
… …
iter1470 value 584.914219
iter1480 value 584.888910
final  value 584.888777
converged
```

The result table is:

Neural Network	Predicted <=50K	Predicted >=50K
Real <=50K	368	88
Real >=50K	48	512
		Test error: 13.38585%

6. Final model and its generalization error

Based on their training error, the final model we choose is a Random Forest with the following parameters:

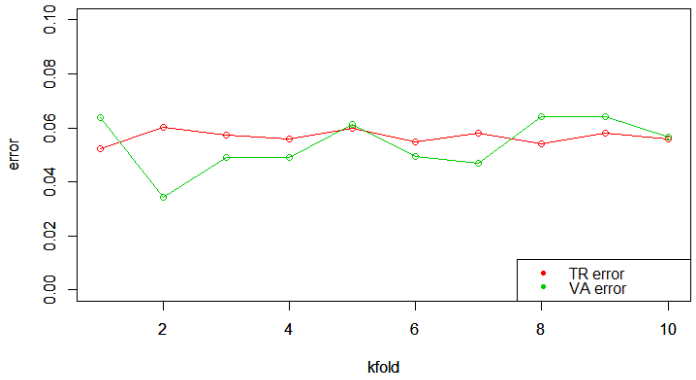
ntree	300
mtry	sqrt(p)

To analysis its generalization error, we need to do two things, first is to check its validation error, another is to check the confidence interval.

First we did a 10-fold Cross-Validation to check the real error of the model. It can represent the validation error of the model, which is much closer to the "real error" of a model than using the "test error".

```
library(TunePareto) # for generateCVRuns()
CV.folds <- generateCVRuns(X.adult.complete$income_class, ntimes=1, nfold=k, stratified=TRUE)
… …
for (j in 1:k) {
   model.randomforest <- randomForest(income_class ~ ., data = X.adult.complete[-va,],
ntree=300, proximity=FALSE)
… … }
```

The result:



We can see that the training error is among 5.5% for all 10-folds, but the validation error varies from 5% to 6% during the 10-folds.

We take the average of the validation error and use it as the true error of our model.

Average of the VA error	5.387734%

Next step we calculate the confidence interval, it's a range of values so defined that there is a specified probability that the value of a parameter lies within it.

We decide to take the 95% confidence interval of our VA error, and use it as the final result of our model.

```
dev <- sqrt(VA.error*(1-VA.error)/N)*1.967
sprintf("(%f,%f)", VA.error-dev,VA.error+dev)
```

The result is

95% CI of the error	(4.69%, 6.08%)

7. Scientific and personal conclusions

The final error with the 95% confidence interval is in the range of (4.69%, 6.08%). This result is much better than the methods mentioned in the second part, and it's mainly because of the data-preprocessing works done before fitting the data to the model.

Another thing is that with many categorical variables in the dataset, the linear classifiers such as Naïve Bayes, SVM with linear kernel don't work well. While the SVM with RBF kernel, random forest performs much better.

8. Possible extensions and known limitations

In the visualization part, I analysis the significant variables that affect the income_class, but I doesn't dig deeper inside.

Future work could be the feature selection based on the visualization or clustering method, which will helps to get a better model.

Another future work is to try some new modeling method, by combining different models together.

References

[1] http://archive.ics.uci.edu/ml/datasets/Adult

[2] Kohavi R. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid[C]//KDD. 1996, 96: 202-207.

[3] Zadrozny, Bianca. "Learning and evaluating classifiers under sample selection bias." Proceedings of the twenty-first international conference on Machine learning. ACM, 2004.

[4] Wang, Haixun, and Philip S. Yu. "SSDT: A scalable subspace-splitting classifier for biased data." Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on. IEEE, 2001.

[5] http://scg.sdsu.edu/dataset-adult_r/