```
library(doParallel) # for speed up
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
# return number of cores in your computer, but keep 2 core free
no_cores <- detectCores() - 2
cl <- makeCluster(no_cores, type="PSOCK")
# start parallel processing
registerDoParallel(cl)
```

```
# Import Data ------------------------------------------------------------------------------------------
# path to folder that holds multiple .csv files (8 csv)
folder <- "C:/Users/user/Desktop/Data_Science/HK_Income/"
# create list of all .csv files in folder
file_list <- list.files(path=folder, pattern="*.csv")
# read in each .csv file in file_list and create a data frame with the same name as the .csv file
for (i in 1:length(file_list)){
  assign(file_list[i], read.csv(paste(folder, file_list[i], sep='')))
}
file_list
```

```
## [1] "hh2018q1_20pct.csv" "hh2018q2_20pct.csv" "hh2018q3_20pct.csv"
## [4] "hh2018q4_20pct.csv" "pp2018q1_20pct.csv" "pp2018q2_20pct.csv"
## [7] "pp2018q3_20pct.csv" "pp2018q4_20pct.csv"
```

```
# combine household income dataframe by rows
hh_2018 <- do.call("rbind", list(hh2018q1_20pct.csv, hh2018q2_20pct.csv, hh2018q3_20pct.csv, hh2018q4_20pct.csv))
# combine personal information dataframe by rows
pp_2018 <- do.call("rbind", list(pp2018q1_20pct.csv, pp2018q2_20pct.csv, pp2018q3_20pct.csv, pp2018q4_20pct.csv))
# merge hh & pp dataframe
hh_pp_2018 <- merge(x=hh_2018,y=pp_2018,by=c("Year_Quarter","Household_reference_no"),all.x = TRUE)
# drop useless Grossing_up_factor
hh_pp_2018$Grossing_up_factor.x <- NULL
hh_pp_2018$Grossing_up_factor.y <- NULL
# Worked_hours have too many level (91 levels), combine it
hh_pp_2018$Worked_hours <- as.integer(hh_pp_2018$Worked_hours)
hh_pp_2018$Worked_hours <- floor(hh_pp_2018$Worked_hours/10)
# display data structure
str(hh_pp_2018)
```

```
## 'data.frame':    46734 obs. of  26 variables:
##  $ Year_Quarter                    : int  20181 20181 20181 20181 20181 20181 20181 20181 20181 20181 ...
##  $ Household_reference_no          : num  1.1e+09 1.1e+09 1.1e+09 1.1e+09 1.1e+09 ...
##  $ Type_of_housing                 : int  3 3 3 3 3 2 3 3 3 1 ...
##  $ Tenure_of_accommodation         : int  3 3 3 3 3 2 1 1 1 3 ...
##  $ Number_of_persons_usually_living: int  3 3 3 2 2 1 1 2 2 2 ...
##  $ Monthly_rent                    : int  9 9 9 8 8 99 99 99 99 2 ...
##  $ Monthly_household_income        : int  10 10 10 9 9 3 10 11 11 3 ...
##  $ Relationship_to_household_head  : int  5 1 7 1 2 1 1 1 2 1 ...
##  $ Age                             : int  7 8 7 9 8 14 12 9 10 12 ...
##  $ Sex                             : int  1 2 2 1 2 1 1 2 1 1 ...
##  $ Educational_attainment          : int  6 7 6 7 8 3 8 7 4 2 ...
##  $ Marital_status                  : int  2 1 2 2 2 2 3 2 2 2 ...
##  $ Economic_activity_status        : int  1 1 8 1 1 6 1 1 1 1 ...
##  $ Whether_being_underemployed     : int  2 2 9 2 2 9 2 2 2 2 ...
##  $ Usual_place_of_work             : int  1 1 9 1 1 9 1 1 1 1 ...
##  $ Industry_for_employed           : int  5 18 99 22 22 99 18 13 18 8 ...
##  $ Industry_for_underemployed      : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ Pre_industry_for_unemployed     : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ Occupation_for_employed         : int  3 3 99 3 3 99 1 3 3 8 ...
##  $ Pre_occupation_for_unemployed   : int  99 99 99 99 99 99 99 99 99 99 ...
##  $ Worked_hours                    : num  7 3 99 3 4 99 5 4 4 4 ...
##  $ M_e_earnings_for_employed       : int  14 14 99 12 13 99 19 14 17 6 ...
##  $ M_e_earnings_for_underemployed  : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ Duration_of_unemployment        : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ Reasons_leaving_for_unemployed  : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ Whether_foreign_domestic_helper : int  2 2 9 2 2 9 2 2 2 2 ...
```

```
# Convert variables type from int to factor
nocol <- ncol(hh_pp_2018) # return number of columns
for (i in 3:nocol){
  hh_pp_2018[,i] <- as.factor(hh_pp_2018[,i])
}
str(hh_pp_2018)
```

```
## 'data.frame':    46734 obs. of  26 variables:
##  $ Year_Quarter                    : int  20181 20181 20181 20181 20181 20181 20181 20181 20181 20181 ...
##  $ Household_reference_no          : num  1.1e+09 1.1e+09 1.1e+09 1.1e+09 1.1e+09 ...
##  $ Type_of_housing                 : Factor w/ 4 levels "1","2","3","4": 3 3 3 3 3 2 3 3 3 1 ...
##  $ Tenure_of_accommodation         : Factor w/ 6 levels "1","2","3","4",..: 3 3 3 3 3 2 1 1 1 3 ...
##  $ Number_of_persons_usually_living: Factor w/ 14 levels "1","2","3","4",..: 3 3 3 2 2 1 1 2 2 2 ...
##  $ Monthly_rent                    : Factor w/ 13 levels "1","2","3","4",..: 9 9 9 8 8 13 13 13 13 2 ...
##  $ Monthly_household_income        : Factor w/ 11 levels "1","2","3","4",..: 10 10 10 9 9 3 10 11 11 3 ...
##  $ Relationship_to_household_head  : Factor w/ 15 levels "1","2","3","4",..: 5 1 7 1 2 1 1 1 2 1 ...
##  $ Age                             : Factor w/ 14 levels "1","2","3","4",..: 7 8 7 9 8 14 12 9 10 12 ...
##  $ Sex                             : Factor w/ 2 levels "1","2": 1 2 2 1 2 1 1 2 1 1 ...
##  $ Educational_attainment          : Factor w/ 8 levels "1","2","3","4",..: 6 7 6 7 8 3 8 7 4 2 ...
##  $ Marital_status                  : Factor w/ 3 levels "1","2","3": 2 1 2 2 2 2 3 2 2 2 ...
##  $ Economic_activity_status        : Factor w/ 8 levels "1","2","3","4",..: 1 1 8 1 1 6 1 1 1 1 ...
##  $ Whether_being_underemployed     : Factor w/ 3 levels "1","2","9": 2 2 3 2 2 3 2 2 2 2 ...
##  $ Usual_place_of_work             : Factor w/ 3 levels "1","2","9": 1 1 3 1 1 3 1 1 1 1 ...
##  $ Industry_for_employed           : Factor w/ 24 levels "1","2","3","4",..: 5 18 24 22 22 24 18 13 18 8 ...
##  $ Industry_for_underemployed      : Factor w/ 8 levels "1","2","3","4",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ Pre_industry_for_unemployed     : Factor w/ 9 levels "1","2","3","4",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ Occupation_for_employed         : Factor w/ 10 levels "1","2","3","4",..: 3 3 10 3 3 10 1 3 3 8 ...
##  $ Pre_occupation_for_unemployed   : Factor w/ 11 levels "1","2","3","4",..: 11 11 11 11 11 11 11 11 11 11 ...
##  $ Worked_hours                    : Factor w/ 11 levels "0","1","2","3",..: 8 4 11 4 5 11 6 5 5 5 ...
##  $ M_e_earnings_for_employed       : Factor w/ 21 levels "1","2","3","4",..: 14 14 21 12 13 21 19 14 17 6 ...
##  $ M_e_earnings_for_underemployed  : Factor w/ 9 levels "1","2","3","4",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ Duration_of_unemployment        : Factor w/ 6 levels "1","2","3","4",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ Reasons_leaving_for_unemployed  : Factor w/ 4 levels "1","2","3","9": 4 4 4 4 4 4 4 4 4 4 ...
##  $ Whether_foreign_domestic_helper : Factor w/ 3 levels "1","2","9": 2 2 3 2 2 3 2 2 2 2 ...
```

```
# Cross-vaildation setting (2 fold CV)-----------------------------------------------------------------
require(caret)
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
require(e1071)
```

```
## Loading required package: e1071
```

```
set.seed(12345)
control <- trainControl(method="repeatedcv", number=2, repeats=2)

# Classification model(1) Multinomial Logistic Regression (MNL)-------------------------------------
multinom_model <- train(Monthly_household_income ~.-Year_Quarter-Household_reference_no,
                data=hh_pp_2018,
                method="multinom",
                MaxNWts = 3000,
                metric = "Accuracy",
                trControl = control,
                trace = FALSE)
# Model summary
multinom_model
```

```
## Penalized Multinomial Regression
##
## 46734 samples
##    25 predictor
##    11 classes: '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11'
##
## No pre-processing
## Resampling: Cross-Validated (2 fold, repeated 2 times)
## Summary of sample sizes: 23369, 23365, 23367, 23367
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.4161958  0.2701590
##   1e-04  0.4162814  0.2702755
##   1e-01  0.4171909  0.2695507
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was decay = 0.1.
```

```
# Fit MNL in data (raw refer to return the predicted income group instead of prob)
hh_qq_MNL <- hh_pp_2018
hh_qq_MNL$Pre_Income_MNL <- predict(multinom_model, newdata = hh_qq_MNL, "raw")
# Measure model performance
confusionMatrix(hh_qq_MNL$Monthly_household_income, hh_qq_MNL$Pre_Income_MNL)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     1     2     3     4     5     6     7     8     9    10
##         1    785   117     8     6    97    42    11     0    40     1
##         2    243   457    23    29   168    25     7     0    37     1
##         3    253   133   101    28   282    48     8     2    56     0
##         4    180    57    60   212   387   129    38     2    67     0
##         5    288    65    54   109  1188   472   152    32   540    11
##         6    220    23    33    84   650   965   171    31   865    20
##         7    139    18    26    53   521   465   587    39  1061    71
##         8     73     4    11    22   298   342   240   177   954    61
##         9     88     6    16    20   320   481   386   130  2133   136
##         10    74     3     3     6   112   158   182    69  1048   226
##         11    70     9     9     5    60   110   135    70   992   213
##           Reference
## Prediction    11
##         1    146
##         2    124
##         3    112
##         4     96
##         5    485
##         6    767
##         7   1245
##         8   1374
##         9   3228
##         10  3389
##         11 13223
##
## Overall Statistics
##
##                Accuracy : 0.4291
##                  95% CI : (0.4246, 0.4336)
##     No Information Rate : 0.5176
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2769
##
##  Mcnemar's Test P-Value : <2e-16
```

```
##
## Statistics by Class:
##
##                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
## Sensitivity          0.32532 0.512332 0.293605 0.369338  0.29096  0.29812
## Specificity          0.98944 0.985668 0.980125 0.977990  0.94823  0.93416
## Pos Pred Value       0.62650 0.410233 0.098729 0.172638  0.34982  0.25202
## Neg Pred Value       0.96420 0.990465 0.994684 0.992045  0.93320  0.94705
## Prevalence           0.05163 0.019087 0.007361 0.012282  0.08737  0.06926
## Detection Rate       0.01680 0.009779 0.002161 0.004536  0.02542  0.02065
## Detection Prevalence 0.02681 0.023837 0.021890 0.026276  0.07267  0.08193
## Balanced Accuracy    0.65738 0.749000 0.636865 0.673664  0.61960  0.61614
##                     Class: 7 Class: 8 Class: 9 Class: 10 Class: 11
## Sensitivity          0.30621 0.320652  0.27371  0.305405    0.5467
## Specificity          0.91883 0.926833  0.87645  0.890334    0.9258
## Pos Pred Value       0.13893 0.049775  0.30717  0.042884    0.8877
## Neg Pred Value       0.96871 0.991315  0.85775  0.987604    0.6556
## Prevalence           0.04102 0.011812  0.16675  0.015834    0.5176
## Detection Rate       0.01256 0.003787  0.04564  0.004836    0.2829
## Detection Prevalence 0.09041 0.076090  0.14859  0.112766    0.3187
## Balanced Accuracy    0.61252 0.623743  0.57508  0.597869    0.7362
```

```
# end parallel processing -------------------------------------------------------------------------
stopCluster(cl)
registerDoSEQ()
```