Predictive modelling

George, John & William

11/17/2021

Data Preparation and cleaning

Getting the missing values percentages

```
missing_rate
```

```
##                    Variable Number_Missing Missing_Rate
## 1                enrollment           7087     56.53318
## 2                employment           7087     56.53318
## 3           employment_type           9448     75.36694
## 4            weekly_work_hrs           9448     75.36694
## 5                 ethnicity           7361     58.71889
## 6                    gender           7383     58.89438
## 7              total_income           7361     58.71889
## 8            academic_level           7361     58.71889
## 9            college/school           7361     58.71889
## 10            mode_transport           7361     58.71889
## 11    transport_reliability           7361     58.71889
## 12             living_alone           7361     58.71889
## 13               dependents           7819     62.37237
## 14           household_head           7417     59.16560
## 15                residence           7441     59.35705
## 16        permanent_address           7441     59.35705
## 17    spent_night_elsewhere          12282     97.97384
## 18 know_homelessness_studt           7476     59.63625
## 19              federal_aid           7476     59.63625
## 20                   FI_q26           7518     59.97128
## 21                   FI_q27           7518     59.97128
## 22                   FI_q28           7518     59.97128
## 23                   FI_q30           7538     60.13082
## 24                   FI_q31           7538     60.13082
## 25     expenditures_changed           7599     60.61742
## 26           income_changed           7599     60.61742
## 27          fed_aid_changed           7599     60.61742
## 28             debt_changed           7599     60.61742
```

The table above displays the number of missing values with missing percentages, from the output its clear that all variables have some amount of missing observation given they are all above 50%, hence a necessary missing data treatment is required.

Table 1: Table : Missing values table displaying percentages

Variable	Number_Missing	Missing_Rate
enrollment	7087	56.53318
employment	7087	56.53318
employment_type	9448	75.36694
weekly_work_hrs	9448	75.36694
ethnicity	7361	58.71889
gender	7383	58.89438
total_income	7361	58.71889
academic_level	7361	58.71889
college/school	7361	58.71889
mode_transport	7361	58.71889
transport_reliability	7361	58.71889
living_alone	7361	58.71889
dependents	7819	62.37237
household_head	7417	59.16560
residence	7441	59.35705
permanent_address	7441	59.35705
spent_night_elsewhere	12282	97.97384
know_homelessness_studt	7476	59.63625
federal_aid	7476	59.63625
FI_q26	7518	59.97128
FI_q27	7518	59.97128
FI_q28	7518	59.97128
FI_q30	7538	60.13082
FI_q31	7538	60.13082
expenditures_changed	7599	60.61742
income_changed	7599	60.61742
fed_aid_changed	7599	60.61742
debt_changed	7599	60.61742

```
## [1] 4998   18
```

The response variables also contains missing values, hence we filter all missing observation with respect to each response variable out and used the remaining data set for further imputation and analysis analysis.

missing value treatment

Method I

The mice package aided in imputation the missing values in the predictor variables, specifically by using the median, mice was chosen because it is robust to data and its imputation style.

Treating imbalance classification

```
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
data_pad_balance<-ovun.sample(FI_q31 ~ ., data = data_pad, method = "both", p=0.5,
dim(data_pad_balance)
```

```
## [1] 4998   18
```

After an EDA on the selected responses variables, highly imbalanced classification was encountered. The imbalanced classification was treated with both sampling methods under the ROSE package.

Converting predictors to category

Partitioning data set

```
## [1] 3350   18
```

```
## [1] 1648   18
```

For the purpose of training and validation of each derived model ,and the estimating of the performance metrics, the entire data set was partitioned in training and testing in a ratio of 2/3 and 1/3 respectively.

Model fitting

Given our data set and its structure, the following supervised classification machine learning algorithm were employed to obtained a predictive model for each given response variable;

```r
#------ Model building -----------
# Create a wrapper function to abstract away the common aspects of model fitting
formula<- FI_q31~.
fit.model <- function(method, tunegrid="", data=NULL, formula=NULL) {

  data <- training
  if(is.null(formula)) formula<- FI_q31~.

  # Train the model
   train(
           formula,
           data = data,
           method = method,
           trControl = trainControl(method = "cv", 5),
           preProcess = c("center","scale"),
           tuneGrid = tunegrid)

}
```

```r
# Logistic Regression
log <-train(formula,
                 data=training,
                 method="glm",
                 family = binomial(link = "logit"),
                trControl = trainControl(method = "cv", 5),
                preProcess = c("center", "scale"))
```

```r
# LDA
lda <- train(formula,
                 data=training,
                 method="lda",
                trControl = trainControl(method = "cv", 5),
                preProcess = c("center", "scale"))
```

```r
#--------------- Elastic Net Models -------------------
# fit a LASSO model
lasso <- fit.model("glmnet", expand.grid(.alpha=1, .lambda=seq(0,0.1,0.01)))

# Fit a Ridge regression model
ridge <- fit.model("glmnet", expand.grid(.alpha=0, .lambda=seq(0,0.1,0.01)))
```

```r
# Bagging
# bag <- fit.model("rf", data.frame(mtry=11))
bag <- train(formula,
                 data=training,
                 method="rf",
                trControl = trainControl(method = "cv", 5),
                preProcess = c("center", "scale"),
                tuneGrid = data.frame(mtry=11),
                ntree = 1000)
```

```r
# Random Forest
# rf <- fit.model("rf", data.frame(mtry=1:10))
# rf <- train(formula,
#                   data=training,
#                   method="rf",
#                  trControl = trainControl(method = "cv", 5),
#                  preProcess = c("center", "scale"),
#                  tuneGrid = data.frame(mtry=1:10),
#                  ntree = 1000)
```

```r
#---------------

# Support Vector Machine with linear kernel
set.seed(125)
trctrl <- trainControl(method = "cv", number=5)
svc <- train(formula, data = training, method = "svmLinear",
                    trControl=trctrl, prob.model=T,
                    tuneLength = 10)
```

```r
# Support Vector Machine with radial kernel
set.seed(125)
trctrl <- trainControl(method = "cv", number=5)
svmR <- train(formula , data = training, method = "svmRadial",
                    trControl=trctrl, prob.model=T,
                    tuneLength = 10)
```

```
## line search fails -1.075911 -0.06488693 1.262876e-05 2.926396e-06 -2.669157e-08 -8.1547e-09 -3.60945

## Warning in method$predict(modelFit = modelFit, newdata = newdata, submodels =
## param): kernlab class prediction calculations failed; returning NAs

## line search fails -1.466557 0.01850916 1.025877e-05 -1.348079e-06 -3.918871e-08 3.881817e-09 -4.0726

## Warning in method$predict(modelFit = modelFit, newdata = newdata, submodels =
## param): kernlab class prediction calculations failed; returning NAs

## line search fails -1.397208 -0.02295924 1.215588e-05 3.470521e-07 -3.957106e-08 -1.268914e-09 -4.814

## Warning in method$predict(modelFit = modelFit, newdata = newdata, submodels =
## param): kernlab class prediction calculations failed; returning NAs

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

Making predictions

Metrics

```
# Create a custom confusion matrix with performance metrics
metrics <- function(model_object, response="", test_data=NULL) {
  # response = "FI_q31"
  # model_object <- log
  #
  if(is.null(test_data)) test_data <- testing

  # make predictions
  prediction <- predict(model_object, test_data)

  target <- test_data[, response]

  cmat <- confusionMatrix(prediction, target, mode = "prec_recall")

  misscal<- round(mean(prediction != target),digits = 2)

 # Returned outputs
 return(list(
   accuracy = (1-misscal),
   mcr = misscal,
   sens = round(cmat$byClass[1],2),
   spec = round(cmat$byClass[2],2),
   fbeta = round(cmat$byClass[7],2)
 ))

}
```

```
metric_log <- metrics(lasso, response = "FI_q31")
#------- Compute performance metrics for the full models ---------------
log.metric <- metrics(log, response = "FI_q31")
lda.metric <- metrics(lda, response = "FI_q31")
# knn.metric <- metrics(knn)
lasso.metric <- metrics(lasso, response = "FI_q31")
ridge.metric <- metrics(ridge, response = "FI_q31")
bag.metric <- metrics(bag, response = "FI_q31")
# rf.metric <- metrics(rf)
svc.metric <- metrics(svc, response = "FI_q31")
# svmP.metric <- metrics(svmP)
svmR.metric <- metrics(svmR, response = "FI_q31")

mod.sum <- data.frame(rbind(
                          c("Logistic", log.metric$mcr, log.metric$accuracy, log.metric$sens, log.metric
                          c("LDA",  lda.metric$mcr, lda.metric$accuracy, lda.metric$sens, lda.metric$spe
                          c("LASSO", lasso.metric$mcr, lasso.metric$accuracy, lasso.metric$sens, lasso.m
                          c("Ridge", ridge.metric$mcr, ridge.metric$accuracy, ridge.metric$sens, ridge.m
                          c("Bagging", bag.metric$mcr, bag.metric$accuracy, bag.metric$sens, bag.metric$
                          c("SVC",  svc.metric$mcr, svc.metric$accuracy, svc.metric$sens, svc.metric$spe
                          c("SVM (Radial Kernel)", svmR.metric$mcr, svmR.metric$accuracy, svmR.metric$se

names(mod.sum) <- c("Model", "Misclassification Rate", "Accuracy", "Sensitivity", "Specificity", "fbeta

kable(mod.sum, align = "lccccc", caption = "Table : Evaluation metrics for Housing Insecurity with Perma
  kable_paper("hover", full_width = F)%>%
```

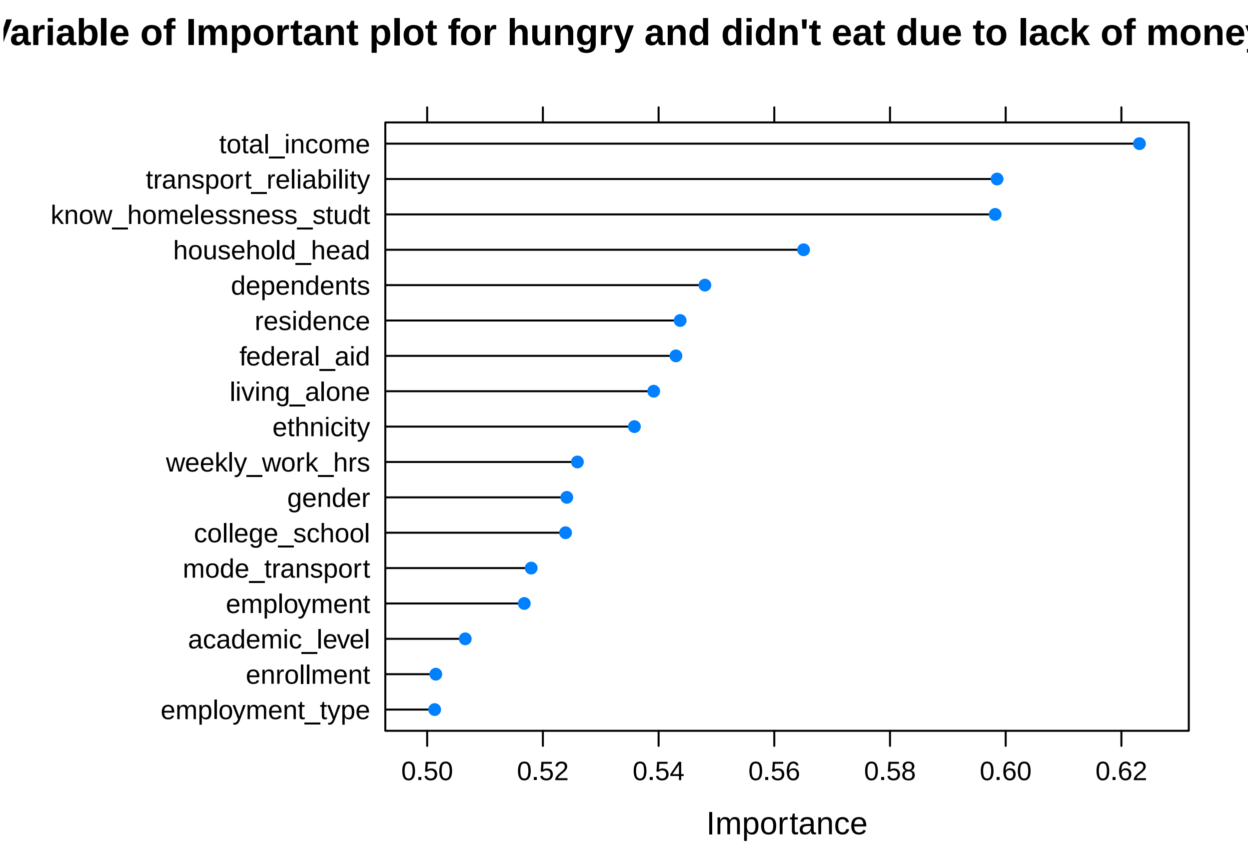
Table 2: Table : Evaluation metrics for Housing Insecurity with Permanent Address as a response

Model	Misclassification Rate	Accuracy	Sensitivity	Specificity	fbeta
Logistic	0.34	0.66	0.63	0.69	0.65
LDA	0.34	0.66	0.62	0.7	0.65
LASSO	0.34	0.66	0.63	0.69	0.65
Ridge	0.35	0.65	0.62	0.69	0.64
Bagging	0.13	0.87	0.9	0.84	0.87
SVC	0.34	0.66	0.63	0.7	0.65
SVM (Radial Kernel)	0.14	0.86	0.88	0.83	0.86

```
        kable_styling(font_size = 12)
```

Base on our table of results a best model is selected base on the performance metrics accuracy, misclassification and fbeta. That is a model with least misclassification , higher accuracy and an fbeta score approaching one. Hence SVM with radial basis function is the best model.

```
Var <- varImp(svmR, scale = FALSE)
plot(Var, main =
       "Variable of Important plot for hungry and didn't eat due to lack of money")
```



The plot above displays the variable of important base on our best model.