OR-568 Applied Predictive Analytics

Costa Rican Household Poverty Level Prediction

Final report

Abstract

Poverty is currently a global threat even in developed economies. Inter-American Development Bank in Costa Rica is aiming to fight poverty using machine learning. The bank aims to ensure that the right people are given the necessary aid. Various observable characteristics like assets in the house, number of unemployed in the house, material of the walls could be used to classify poverty levels. If efficient algorithms are developed to classify poverty based on some observable features of the household, traditional manual methods can be improved and automated.

In this project, we analyze different features, derive new features and explore different machine learning algorithms to predict the poverty of households. Derived features and models performed could improve the performance of traditional methods in locating poverty. These algorithms could be scaled to different countries of the world to assess poverty levels and evaluate the social needs of people under poverty.

Objective

The primary objective is to predict poverty level of Costa Rican people, based on household characteristics. In this process we would be able to find various characteristics which contribute to poverty, evaluate the social needs of people living under poverty, demographic patterns etc.

Dataset Description

The training dataset was provided by Kaggle[1]. This is the main table, which is divided into two files for Train (with a Target column) and Test (without the Target column). One row represents every individual in our data sample. The train file has **9558** rows and **143** columns (with Target) and the test file has **23857** rows and **142** columns.

Target Variable

Poverty levels are categorized into four types. The poverty levels and its respective codes are given below.

1 = extreme poverty
2 = moderate poverty
3 = vulnerable households
4 = non vulnerable households

The frequency counts for each class is visualized with the help of the bar graph. We can observe from the bar graph that the target variable is highly imbalanced with most of the individuals being non-vulnerable.

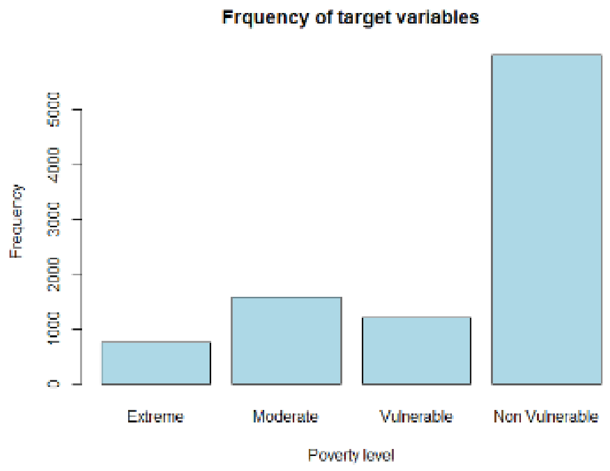


Figure 1 : Frequency of Target Variables

Preprocessing:

Aggregation:

We had to do intense preprocessing for this data. The data had some columns representing individuals and some columns representing the household. For example, age, gender and level of education are columns representing the individuals whereas house rent, number of elders in the house etc are columns representing households. We had to convert those individual level data into household data.

We manually identified all the individual level variables. There were 36 variables representing individuals. These variables were aggregated for household level. For example, age was aggregated into mean, min and max age. Years of education for each individual was converted to mean, min and max years of education for the household etc.

Missing values:

Three variables [rent, number of years remaining to complete schooling, number of tablets owned] had more than 80% of the data missing. Though it would have been appropriate to delete those columns we wanted to find if there is any pattern in those missing values.

House rent:

There was a column which indicates the home ownership status. House ownership status was broadly categorized into 4 categories. Namely, house owned and the debt is paid off, house is owned and debt is being paid, the house is rented and the house is in precarious condition. We wanted to find if there is any pattern between the missing values of the rent column and home ownership column. The frequency count of home ownership status for the individuals for whom the rent information is missing is visualized with the help of the bar graph below.

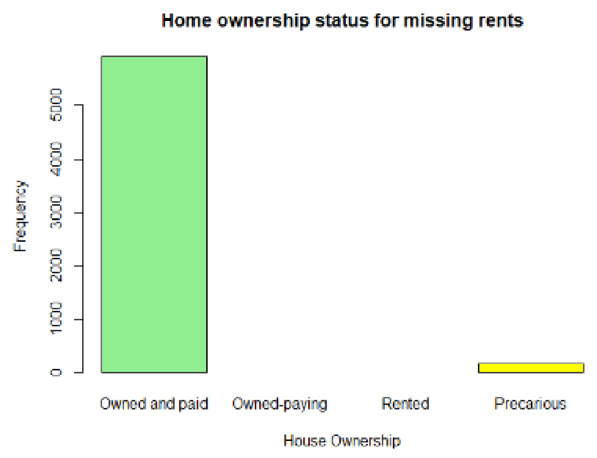
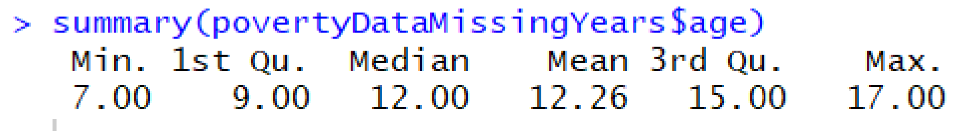


Figure 2 : Bar graph representing frequency count of home ownership

We can observe that the individuals for whom the rent information is missing own their home. Therefore, rent is left blank for people who live in their own house. Missing values in the rent column is substituted with zeros.

Years left in school :

Years left or number of years remaining to complete schooling has more than 80% of data missing. When we found the summary of age for the individuals for whom the years left in school is available, we found that the maximum age for whom the the data is available is just 17. Which means, the people above 17 simply do not go to school. The summary information is given below.



```
> summary(povertyDataMissingYears$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   7.00    9.00   12.00   12.26   15.00   17.00
```

Therefore, missing values are substituted with zero.

Number of tablets owned :

The individuals for whom the number of tablets owned data is missing simply don’t have any tablets. This was verified with the help of a column which indicates weather the household owns any tablets.

<u>Feature Engineering :</u>

We performed some feature engineering to remove correlated features, derive new features, modify existing features etc.
For instance, room rent was calculated by dividing rent by number of rooms, number of persons per room was calculated by dividing number of persons in each household by number of rooms and , number of persons per bedrooms was calculated by dividing number of persons in each household by number of bedrooms. Rent divided by number of males in the house, number of females in the house, number of males under age of 12, number of females under age of 12 were calculated and included as features.

Similarly, various new features were calculated which increased the feature count from 143 to 177. These derived features helped us in improving the metrics of our model to a great extent.

Downsampling :

Downsampling is reducing the records of majority class by sampling the data randomly. We thought that downsampling would alleviate the data imbalance problem but it did not help us increase the accuracy of our models.

Exploratory Data Analysis :

We performed some exploratory data analysis using the features we thought would be important in predicting the poverty levels of the households.

Poverty and television :

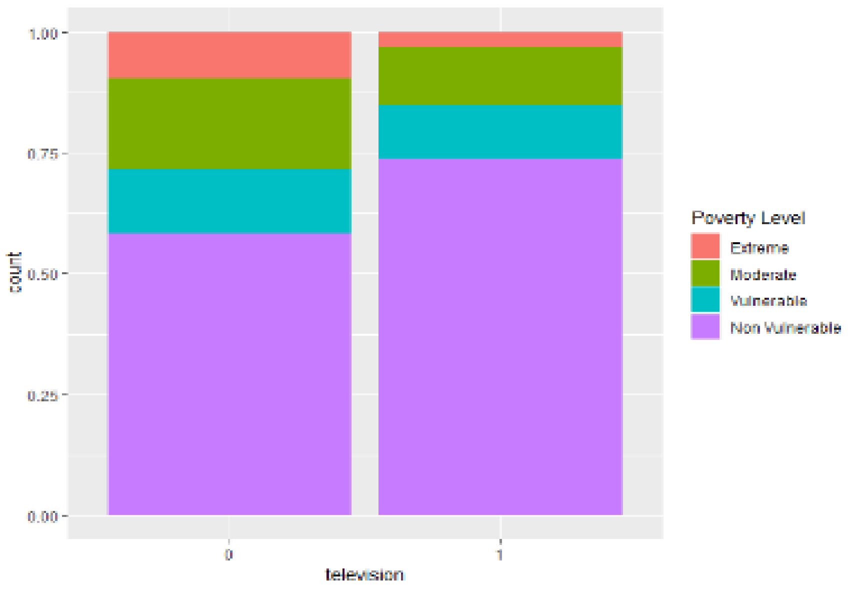


Figure 3 : Chart signifying the poverty level and television counts

In the barchart above, 1 indicates the household owns a television and 0 indicates they don't own any. We can observe a clear distinction in poverty level between number of people who own television and people who don't. It is evident that television is a rare phenomenon in households which are in extreme poverty levels.

Poverty and Computers :

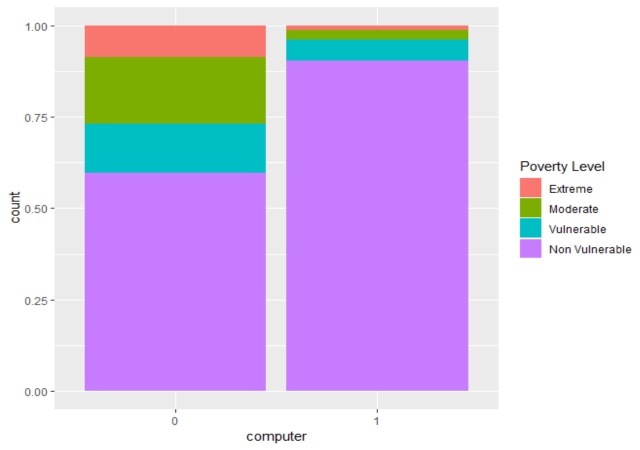


Figure 4 : Chart representing the presence computer in different poverty level

In the barchart above, 1 indicates the household owns a Computer and 0 indicates they don't own any. We can observe that the access to computers are mostly restricted to households which are categorized as non-vulnerable. In the age of internet, it is evident that computers are still a dream to the poor in Costa Rica.

Poverty and Post-Graduate Education :

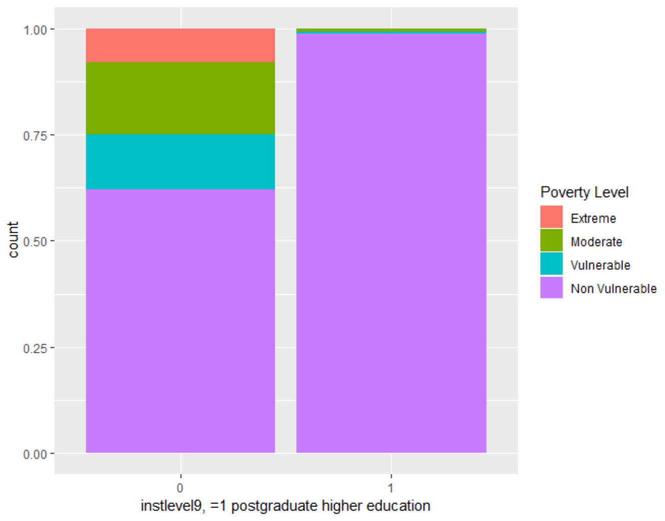


Figure 5 : Chart representing the higher education attendance for different poverty levels

Surprisingly, frequency of people from moderate or extreme poverty levels attending post graduate education is near zero. Postgraduate education is still infeasible for people other than the ones who belong to the non-vulnerable category.

Number of Females of different ages by household type :

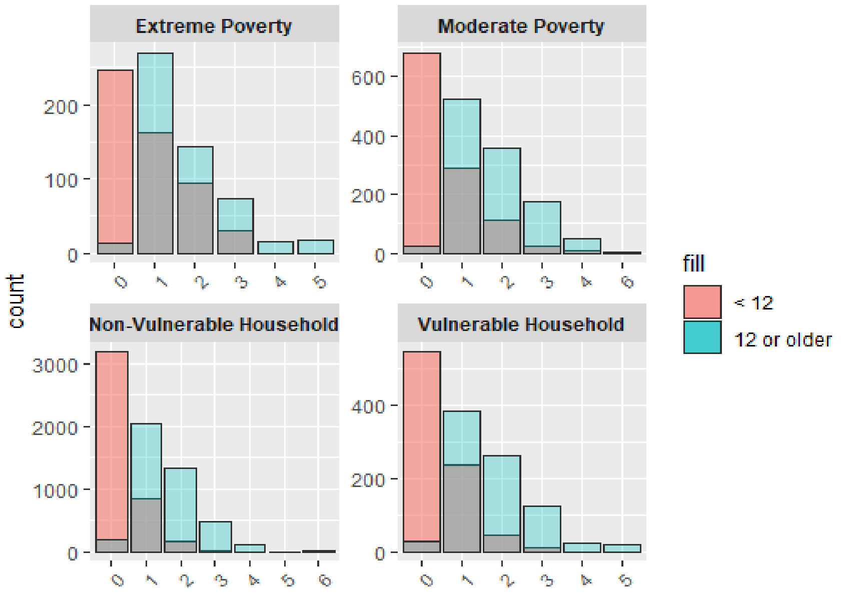


Figure 6: Chart representing the number of Females of different ages by household type

In the above graph, we can say that most of the households do not have number of females younger than age 12. Also most of the households that have the females 12 or older have only one female per household.

Number of males of different age groups for each household type :

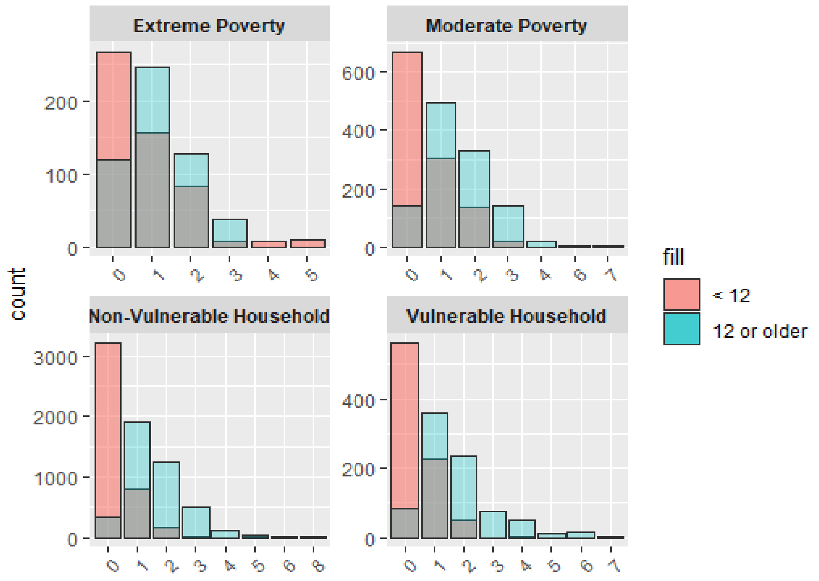


Figure 7: Chart representing the number of males of different age group in each household

Number of males per household can be visualized in the above graph. Similar to the distribution of the females, most of the households do not have Males younger than 12 but one interesting observation from the graphs is in the extreme poverty households which have number of males more than 3 only have males younger than 12 years of age.

Total Number of persons per house for each household type :

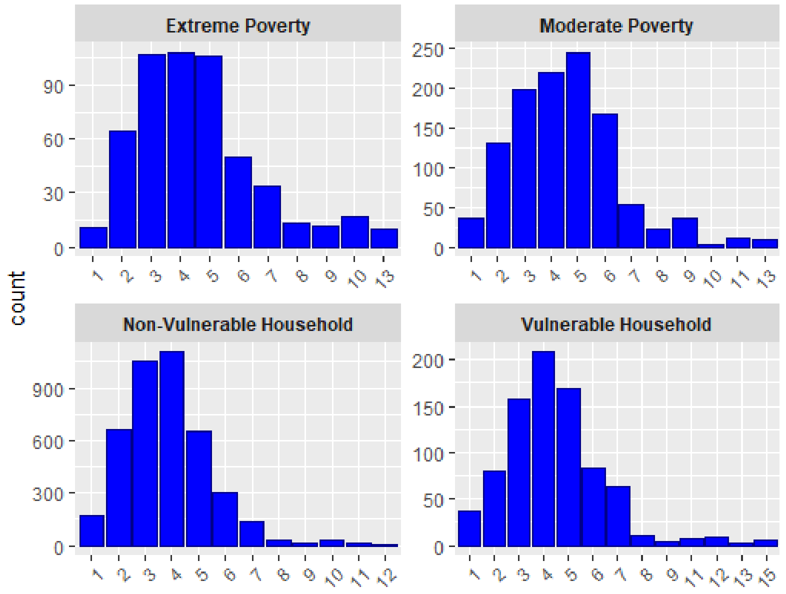


Figure 8: Chart representing total number of persons per house for each poverty type

From the visualizations, we can see that the vulnerable household type has greater total number of members per household. And the most frequent value for the number of members in the household per each level is 4 except for Moderate poverty which is 5.

<u>Models</u> :

Model Metric :

Macro f1-score was used as a metric to evaluate the performance of the models. Macro f1 score is the average of f1 scores of each class in the dataset. This is also used as the evaluation metric used by kaggle. The formula for the macro-f1 score is given below.

$$Macro - F1 = \frac{F1\text{–}Class\text{–}1 + F1\text{–}Class\text{–}2 + F1\text{–}Class\text{–}3 + F1\text{–}Class\text{–}4}{4}$$

<u>SVM Model :</u>

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane, that can be employed for both classification and regression purposes. For our dataset, we use Caret Library and traincontrol method to prep our model. Traincontrol method is generally used to control the computational nuances of the train function. One of the advantages of using SVM is that, it works really well with clear margin of separation whereas on the other hand it does not perform well when there are noises in the data or if the data is huge.

Here we use the Linear kernel which holds the details about resampling method and a tunelength parameter (iterations) of 10. Regardless of the method used, model such as these are effective based on the right parameters that are being used. On doing so, we could see that the accuracy of the model is 0.128. The model does not certainly perform well because the noise and unbiased structure of data makes it difficult for models such as SVM to compute accuracy score.

<u>PCA model :</u>

This method is an unsupervised learning technique and is used to reduce the dimension of the data with minimum loss of information. To reduce the computational time and to capture the variance we performed PCA. The below image is obtained using factoextra R package to generate an interpretation and visualize the PCA. We also use the eigenvalues to determine the number of principal components to be considered.

Contribution of each feature to the PCA could not be visualised because of the sheer number of features. From the plot below a clear distinction between each class is evident with the help of colored ovals.

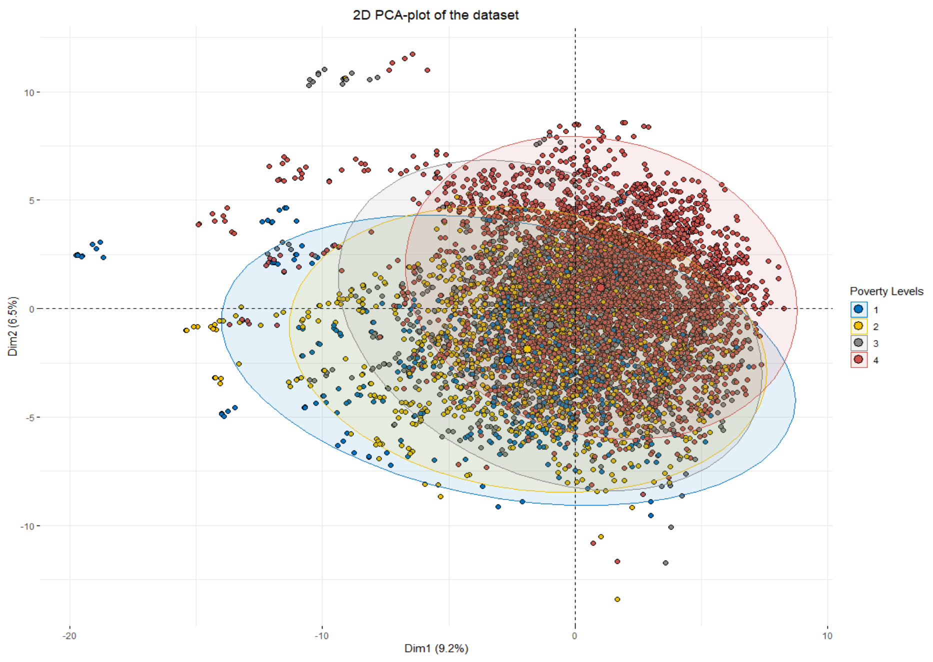


Figure 9 : PCA representing the target variables in a 2D plot

The scree plot indicating different principal components and percentage of variance explained by them is given below. We can observe that only around 18% of variance is explained by the first three principal components.

Though PCA helped us in reducing the dimensionality, we lost considerable information in the process. When we used PCA as features in our machine learning models, accuracy of our models reduced so we did not use features from our models.

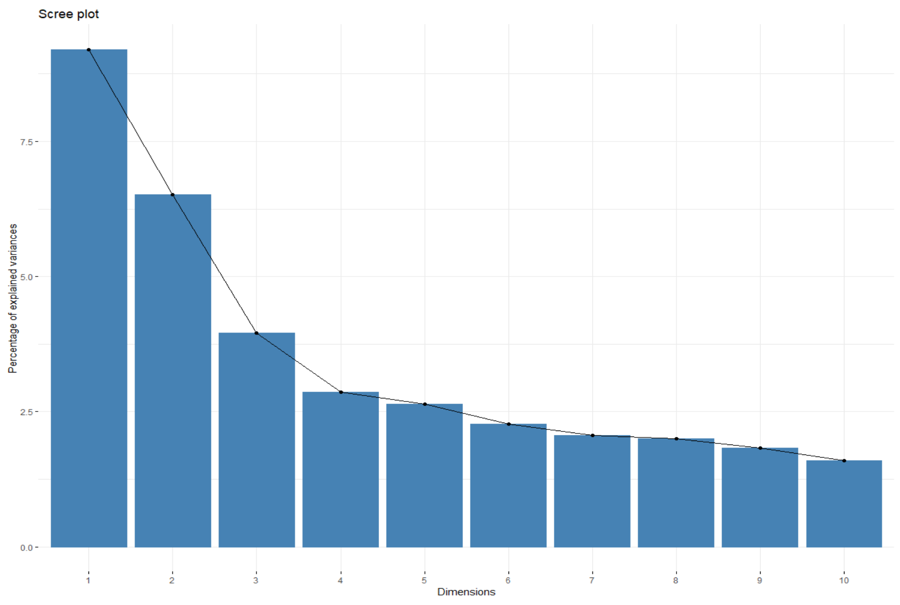


Figure 10: Scree plot displaying the variation proportion

<u>Regularization Models :</u>

To avoid overfitting, we have to regularise the coefficients estimates towards zero or zero, i.e. reducing the number of features so that the model is constrained and doesn't overfit. Thus reducing the complexity of model. This is achieved by introducing a regularisation or penalty term 'lambda(λ)'. The two most popular regularization models are Ridge and Lasso models.

<u>Ridge Regression :</u>

In Ridge regression, the penalty term λ, is simply the squared magnitude of the coefficient. This penalty term can be tuned to achieve the right variance and least MSE. If λ is zero, we get back Ordinary Least Square(OLS) model. If λ is very high, then it leads to under fitting.

<u>Lasso Regression :</u>

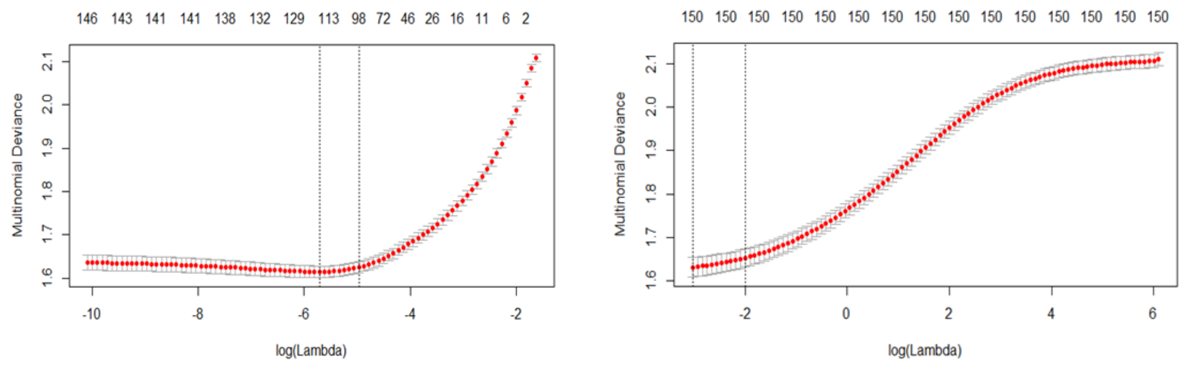
In Lasso regression, the penalty term λ is the absolute value of magnitude of coefficient. Again, if *lambda* is zero then we will get back OLS, whereas very large value will make coefficients zero hence it will under-fit.

The key difference between these techniques is that Lasso shrinks the least important feature's coefficient to zero thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.

Value of the tuning parameter λ:

The tuning parameter controls the amount of regularization, so choosing a good value of the tuning parameter is crucial. Cross-validation is a simple, intuitive way to estimate prediction error.

On implementing the ridge regression, the macro F1 score was as low as 0.142. On implementing the lasso regression, the model was reduced from over 140 predictors to 19 predictors, by the Feature selection property of Lasso Regression. But the macro F1 score was very low with a value of 0.150.



Ridge Regression Lasso Regression

Figure 11: Ridge and Lasso regression implementation

Multinomial Logistic Regression :

Multinomial logistic regression is a parametric model used for predicting multi class level problems. We used Multinomial logistic regression for the data discarding the order information for the target variable i.e., the poverty level. Using the multinom function on the NNet package, we built a multinomial logistic regression model on the data with the poverty level as the target variable and other variables as the predictors. The class level F1 score for the regression is 0.165.

We performed the Wald test for determining the significance of the predictors on the model based on their coefficient values. For few levels, the variables ageSQ and edjefa were shown to be insignificant with a p-value of higher than 0.05. So, we built a next model for the second iteration excluding the 2 variables. The overall (class-level) f-1 score improved by a 3% which is 0.17.

Random Forest :

Random forest is a popular ensemble method that can be used to build the predictive models for Classification and Regression as well. It builds forest which is ensemble of decision trees. In general, it creates multiple decision trees and ensemble them to get more accurate model. Random forest was implemented by using RandomForest library. After tuning hyperparameters, caret library is used.

Macro F1 score is noted as 0.437 and accuracy on the validation set is 0.92.
Tuning hyperparameters, mtry = 150, splitrule = gini and min.node.size = 1

After tuning hyperparameters, With 10 fold cross validation, Macro F1 score is noted as 0.44 and accuracy on the validation set is 0.94. When we used the above model to predict the kaggle's test dataset we received macro f1-score of 0.38

Variable Importance Plot:

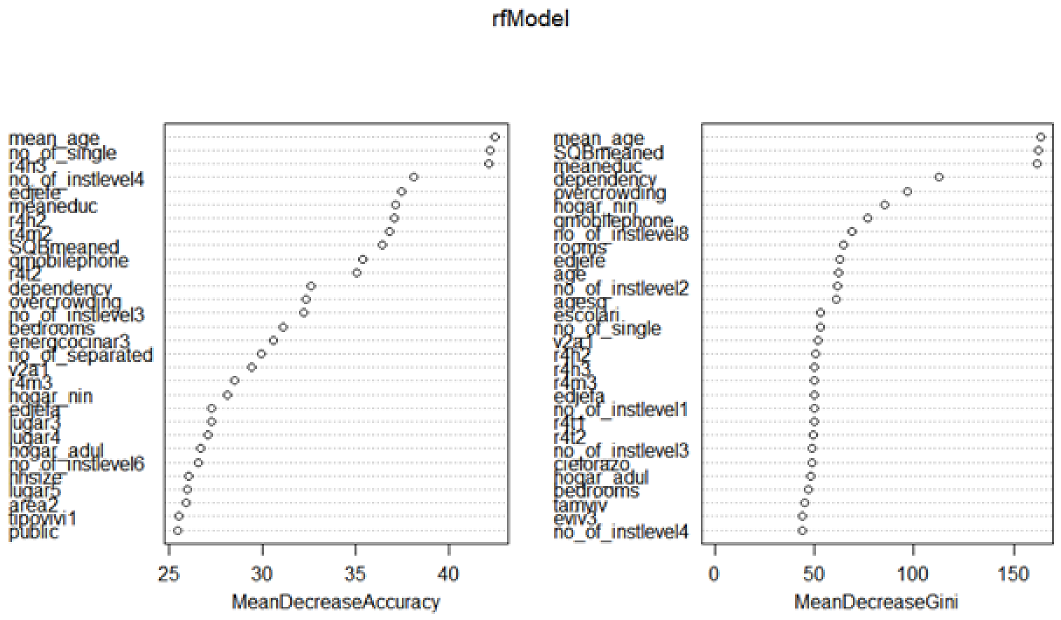


Figure 12: Variable Importance plot representing Mean accuracy and Gini scores

From the Variable importance plot, we can observe that mean age of the household, Number of singles in the household and total number of males in the household are important variables to predict the poverty level of the household.

<u>XGBoost :</u>

XGBoost or extreme gradient boosting is fast and efficient algorithm. It is also the go to algorithm for winners of kaggle competitions.

We performed xgboost by converting our features into matrix and converting labels into series of numbers from 0 to 3 since the xgboost model accepts data in this format.

We performed five fold cross validation to find the right parameters for the XGB model. We obtained a macro f1-score of 0.45 on our test dataset. When the data was downsampled, the model did not perform well and we received a macro f1-score of 0.38.

When we used the above model to predict the kaggle’s test dataset we received macro f1-score of 0.40.

Variable Importance plot:

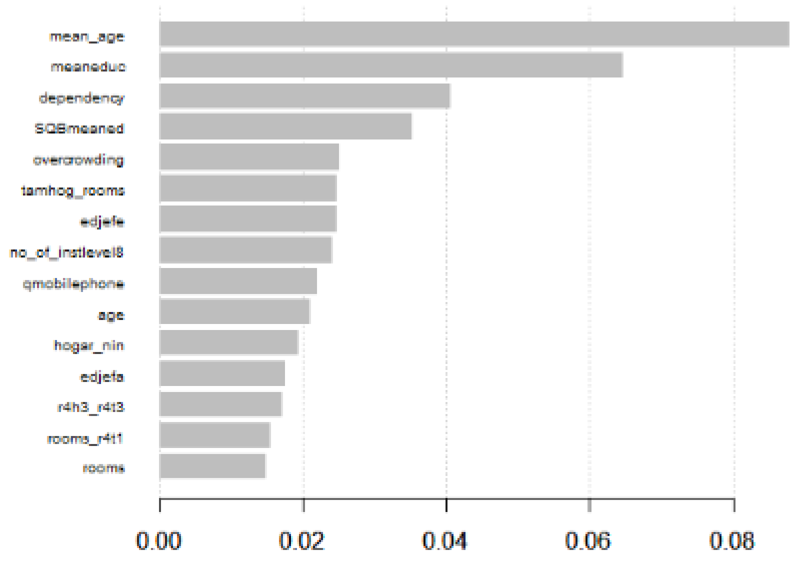


Figure 13: Variable importance plot of the top 15 variables

From the variable importance plot, we can observe that there is not much difference between variable importances obtained from random forest model and XGBoost model. We can also observe that The features which we derived by feature engineering like mean age, mean education have more importance than the features which were originally present is the dataset.

<u>Step by Step instructions on how to run the code:</u>

Step 1: Install latest version of R
Step 2: Install latest version of R Studio.
Step 3: Set the working directory path within both the attached R codes.
Step 4: Download the dataset (from the blackboard) in the working directory of the project.
Step 5: Run both the R code files.

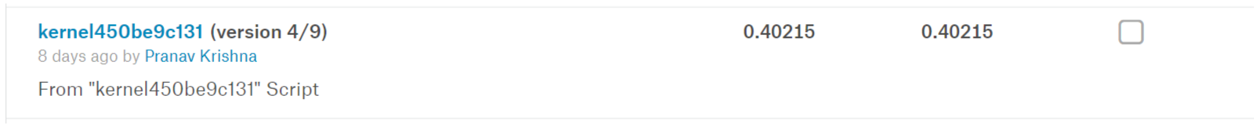
Results from our validation set :

Model	**F1 score**
SVM	0.128
maboost	0.103
Random Forest	0.44
XGBoost	0.45
Multinomial Logistic Regression	0.17
Ridge	0.14
Lasso	0.15

Results from kaggle's test set :

Model	F1-Score
XGBoost	0.40
Random Forest	0.37
Kaggle's state of the art model	0.44

XGBoost Model

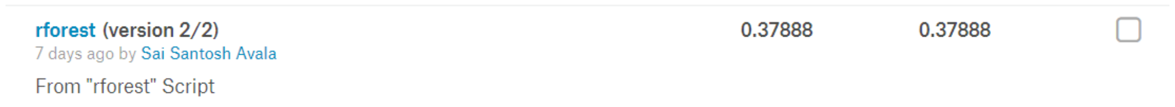


kernel450be9c131 (version 4/9) 0.40215 0.40215

8 days ago by Pranav Krishna

From "kernel450be9c131" Script

Random Forest Model



rforest (version 2/2) 0.37888 0.37888

7 days ago by Sai Santosh Avala

From "rforest" Script

Failed Approaches:

Boosting

We tried various boosting models like maboost and adaboost before finalizing on XGBoost. XGBoost gave superior performance compared to any other boosting algorithm.

Ensembling

A simple voting classifiers require at least 3 good models. We just had random forest and XGBoost giving considerable macro-f1 score so a voting classifier would not have performed well. We tried stacking the prediction results of random forest and xgboost but we had to split the data into training, validation and test. Split of 3 would reduce the data size considerably and performance comparison between models of different test sizes would not be appropriate.

Ensembling XGBoost by Repeated downsampling

Downsampling works by sampling the data randomly. Therefore, a new version of data is created whenever a dataset is downsampled. Repeatedly downsampling and performing a XGBoost model on each of the downsampled data and ensembling those results would improve the accuracy. We tried this method but did not have necessary computational power to successfully complete it. The model was running past the runtime of 2 hrs.

Conclusion :

We tried and succeeded in developing a model which predicts the poverty in each household. We also succeeded in finding some of the driving forces of poverty in Costa Rica. We could conclude that Lack of access to computers, televisions and higher education could prove as vital information for Inter-American Development Bank as well as the government of costa rica in fighting poverty.

References :

[1] Cock , D. (2011). Costa Rica Housing dataset. Retrieved from Kaggle: House Prices: Advanced Regression Techniques: https://www.kaggle.com/c/costa-rican-household-poverty-prediction/data
[2] Earth Trends (2003). "Biodiversity and Protected Areas – Costa Rica" (PDF). World Resources Institute. Archived from the original (PDF) on 27 September 2011. Retrieved 8 June 2008.
[3] Dickerson, Marla; Kimitch, Rebecca (23 March 2006). "Costa Rica Seeks to Shut Its Doors to Illegal Migrants From Nicaragua". Los Angeles Times. Retrieved 2 May 2010
[4] Ramsey, F. L., & Schafer, D. W. (2013). *The statistical sleuth: A course in methods of data analysis*. Boston: Brooks/Cole, Cengage Learning.
[5] Leif E. Peterson, 2010. "MLOGITROC: Stata module to calculate multiclass ROC Curves and AUC from Multinomial Logistic Regression," Statistical Software Components S457181, Boston College Department of Economics.
[6] Ahmed, AU & Bouis, HE 2002, Weighing What's Practical: Proxy Means Tests for Targeting Food Subsidies in Egypt. Food Consumption and Nutrition Division, IFPRI, Washington DC, United States.

Appendix A: R Codes

1. All models except XGBoost

```
################################ Poverty Prediction ########################################

if (!require("randomForest")) install.packages("randomForest")
if (!require("dplyr")) install.packages("dplyr")
if (!require("tidyverse")) install.packages("tidyverse")
if (!require("xgboost")) install.packages("xgboost")
if (!require("corrplot")) install.packages("corrplot")
if (!require("glmnet")) install.packages("glmnet")
if (!require("rpart")) install.packages("rpart")
if (!require("factoextra")) install.packages("factoextra")
if (!require("gridExtra")) install.packages("gridExtra")
if (!require("maboost")) install.packages("maboost")
if (!require("pROC")) install.packages("pROC")
if (!require("ROCR")) install.packages("ROCR")
if (!require("SDMTools")) install.packages("SDMTools")
if (!require("Hmisc")) install.packages("Hmisc")
if (!require("nnet")) install.packages("nnet")
if (!require("caret")) install.packages("caret")
if (!require("ISLR")) install.packages("ISLR")

library(randomForest)
library(dplyr)
library(tidyverse)
library(xgboost)
library(glmnet)
library(ISLR)
library(caret)
library(corrplot)
library(nnet)
library(Hmisc)
library(SDMTools)
library(ROCR)
library(pROC)
library(maboost)
library(rpart)
library(gridExtra)
library(factoextra)

#setwd("C:\\Users\\vidhy\\Downloads\\costa-rican-household-poverty-prediction")
povertyData<-read.csv("train.csv")

################################## Data Preprocessing ####################################

colnames(povertyData)[colSums(is.na(povertyData)) > 0]
```

```
sapply(povertyData, function(x) sum(is.na(x)))[colSums(is.na(povertyData)) > 0]
nullValueCols<-sapply(povertyData, function(x) sum(is.na(x)))

#### 1. Treating Null values
#### House -Rent
povertyDataV2Na <- povertyData[rowSums(is.na(povertyData['v2a1'])) > 0,]
houseOwnership=c(table(povertyDataV2Na$tipovivi1)[2],table(povertyDataV2Na$tipovivi2)[2],table(poverty
DataV2Na$tipovivi3)[2],table(povertyDataV2Na$tipovivi4)[2])
barplot(houseOwnership,names.arg=c("Owned and paid","Owned-paying","Rented","Precarious"),
        col=c("light green","red","green","yellow"),las=0.5,main="Home ownership status for missing
rents",xlab="House Ownership",ylab = "Frequency")

#### Number of tablets
povertyDataV18Na <- povertyData[rowSums(is.na(povertyData['v18q1'])) > 0,]
tabletOwnership=c(table(povertyDataV18Na$v18q))

#### Years behind in school

povertyDataVrez <- povertyData[rowSums(is.na(povertyData['rez_esc'])) == 0,]
summary(povertyDataVrez$age)

#### Convet all NAs to zeors
povertyData[is.na(povertyData)] <- 0

#### Check if all nulls are trated
colnames(povertyData)[colSums(is.na(povertyData)) > 0]

############################Feature Engineering####################################
#Removing all columns which are squared
povertyData <- povertyData[,-(134:140),drop=FALSE]
#povertyData <- povertyData[ , -which(names(povertyData) %in% c("female"))]

has_many_values <- function(x) n_distinct(x) > 1
dup_var <- function(x) lapply(x, c) %>% duplicated %>% which

##################################################################################


#Aggregate individual variable to household level
povertyData <- povertyData %>%
  group_by(idhogar) %>%
  mutate(mean_age = mean(age, na.rm = TRUE)) %>%
  mutate(no_of_disabled = sum(dis)) %>%
  mutate(no_of_children = sum(estadocivil1)) %>%
  mutate(no_of_coupledunion = sum(estadocivil2)) %>%
  mutate(no_of_married = sum(estadocivil3)) %>%
  mutate(no_of_divorced = sum(estadocivil4)) %>%
  mutate(no_of_separated = sum(estadocivil5)) %>%
  mutate(no_of_widower = sum(estadocivil6)) %>%
  mutate(no_of_single = sum(estadocivil7)) %>%
  mutate(no_of_instlevel1 = sum(instlevel1)) %>%
  mutate(no_of_instlevel2 = sum(instlevel2)) %>%
  mutate(no_of_instlevel3 = sum(instlevel3)) %>%
  mutate(no_of_instlevel4 = sum(instlevel4)) %>%
  mutate(no_of_instlevel5 = sum(instlevel5)) %>%
  mutate(no_of_instlevel6 = sum(instlevel6)) %>%
  mutate(no_of_instlevel7 = sum(instlevel7)) %>%
  mutate(no_of_instlevel8 = sum(instlevel8)) %>%
  mutate(no_of_instlevel9 = sum(instlevel9)) %>%
  ungroup()

povertyData <- povertyData %>%
  #select(-tamviv) %>% # number of persons living in the household
  #select(-hogar_total) %>% # # of total individuals in the household
  #select(-r4t3) %>% # Total persons in the household
  #select(-tamhog) %>% # size of the household
  #select(-r4t1) %>% # persons younger than 12 years of age
  #select(-r4t2) %>% # persons 12 years of age and older
  #select(-agesq) %>% # Age squared
  select(-Id) %>% # removing id
  select(-idhogar)

#### Check if the dataset has any non numeric column(s)
povertyData %>%
  select_if(funs(!is.numeric(.)))
#### Recode values in dependency, edjefe, edjefa
povertyData[,c("dependency","edjefe","edjefa")] <- povertyData %>%
  select(dependency,edjefe,edjefa) %>%
  mutate_all(funs(ifelse(. == "yes",1,ifelse(. == "no",0,.)))) %>%
  mutate_all(as.numeric)
#### Train and test split
row<-nrow(povertyData)
set.seed(12345)
trainindex <- sample(row, row*.7, replace=FALSE)
training <- povertyData[trainindex, ]
validation <- povertyData[-trainindex, ]

train_labels <- as.numeric(training$Target) - 1
test_labels<-as.numeric(validation$Target)-1
train_data <- as.matrix(training[,-134])
test_data <- as.matrix(validation[,-134])

################################### Macro F1 score ########################################

f1_score <- function(predicted, expected, positive.class="1") {
  predicted <- factor(as.character(predicted), levels=unique(as.character(expected)))
  expected  <- as.factor(expected)
  cm = as.matrix(table(expected, predicted))

  precision <- diag(cm) / colSums(cm)
  recall <- diag(cm) / rowSums(cm)
  f1 <-  ifelse(precision + recall == 0, 0, 2 * precision * recall / (precision + recall))

#### Assuming that F1 is zero when it's not possible compute it
  f1[is.na(f1)] <- 0

#### Binary F1 or Multi-class macro-averaged F1
  ifelse(nlevels(expected) == 2, f1[positive.class], mean(f1))
}

################################ Ridge and Lasso ########################################

#### Ridge Regression
ridge.mod = glmnet(train_data,train_labels,alpha=0, family="multinomial", type.multinomial="grouped")
#### Plotting the coefficents of all features against different lambda values
plot(ridge.mod,xvar="lambda")

#### cv.glmnet() - performs 10-fold cross-validation
ridge_cv=cv.glmnet(train_data,train_labels,alpha=0, family="multinomial", type.multinomial="grouped")
#### Plotting log(lambda) against multinomial deviance
plot(ridge_cv)

#### Select lamda that minimizes training MSE
bestlam=ridge_cv$lambda.min
bestlam #0.04901


#### making predictions using lambda.min
ridge.predicted<- predict(ridge.mod, s=bestlam, newx= test_data, type='class')
ridgeF1<-f1_score(ridge.predicted,test_labels) #0.146


#### Calculating the confusion matrix for ridge regression
confMat<-confusionMatrix(factor(ridge.predicted), factor(test_labels))
confMat #accuracy -0.6743

mean((as.numeric(ridge.predicted)-as.numeric(test_labels))^2) #1.03


#### The Lasso ( alpha=1)
lasso.mod = glmnet(train_data,train_labels,alpha=1, family="multinomial", type.multinomial="grouped")

#### Plotting the coefficents of all features against different lambda values
plot(lasso.mod, xvar="lambda")
```

```
summary(lasso.mod)

#### using cross validation
lasso_cv=cv.glmnet(train_data,train_labels,alpha=1, family="multinomial", type.multinomial="grouped")

#### Plotting log(lambda) against multinomial deviance
plot(lasso_cv)

bestlam_lasso=cv.out$lambda.min #0.004490
lasso.pred=as.numeric(predict(lasso.mod,s=bestlam, newx= test_data, type='class'))

#### Calculating the confusion matrix for Lasso regression
confMat<-confusionMatrix(factor(lasso.pred), factor(test_labels))
confMat #acc: 65.34%

lassoF1<-f1_score(lasso.pred,test_labels) #0.1505

#### Feature Selection using Lasso
x=model.matrix(Target~.,povertyData)
y=povertyData$Target

out=glmnet(x,y,alpha=1, family= "multinomial",lambda=grid)
#### lasso.coef has features for each target class
lasso.coef=predict(out,type="coefficients",s=bestlam)

################################## MA BOOST ##########################################

#### The maboost and rpart library is used to run the model.
#### Initially the model is fit and predicited against the train dataset.

test1 <- as.data.frame(test_data)
fit_maboost <- maboost(Target ~ ., data=training, iter = 100 ,verbose = TRUE)

#### Predicting on the train set to check for overfitting
predict_maboost <- predict(fit_maboost, training, type = "response")
#### Predicting on the test set
predict_maboost_test <- predict(fit_maboost, test1, type = "response")

head(predict_maboost)
score_maboost <- sum(training$Target == predict_maboost)/nrow(training)
score_maboost #0.6797


#### Calculating Macro F1 score
f1_score(predict_maboost_test,test_labels) #0.106

############################### Principal Component Analysis ##############################
```

```
#### Initially a subset of the dataset is created by removing
#### the variable with missing value causing noise in the dataset.
Data <- subset( training, select = -elimbasu5 )
training1 <- training[-training$elimbasu5]
str(Data)

#### The below code helps us to fetch the correlation plot.
povertyDataCoVar <- cov(povertyData)
corrplot(povertyDataCoVar, method="circle",is.corr = FALSE)

#### eigen() to calculate the eigen values and eigen vectors
target.eigen <- eigen(povertyDataCoVar)
str(target.eigen)

which(apply(training, 2, var)==0)

#### We are using built in functions prcomp which uses spectral decomposition approach.

pcadata1 <- prcomp(Data, center = TRUE, scale = TRUE)
summary(pcadata1)

#### The below plot helps us to visualize the PCA plot between the four target variables
#### showing that there exists a significant difference between them.

fviz_pca_ind(pcadata1, geom.ind = "point", pointshape = 21,
         pointsize = 2,
         fill.ind = as.factor(Data$Target),
         col.ind = "black",
         palette = "jco",
         addEllipses = TRUE,
         label = "var",
         col.var = "black",
         repel = TRUE,
         legend.title = "Poverty Levels") +
  ggtitle("2D PCA-plot of the dataset") +
  theme(plot.title = element_text(hjust = 0.5))

############################## Support vector Machines ####################################

#### traincontrol() - to control the computational nuances present in the train dataset
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

#### The below SVM method uses Linear kernel. Preprocess function uses preprocess
#### parameter - center and scale and the tunelength which holds an integer value.

svm_Linear <- train(as.factor(Target) ~., data = Data, method = "svmLinear",
                    trControl=trctrl,
                    preProcess = c("center", "scale"),
                    tuneLength = 10)


test_pred <- predict(svm_Linear, newdata = test_data)

#### Macro f1 score - 0.128
f1_score(test_pred,test_labels)

############################ Multinomial Logistic Regression ##############################

trainingSubset <- subset( training, select = -elimbasu5 )
validationSubset <- subset( validation, select = -elimbasu5 )

#### Building the multinomial logistic model with Target as y and remaining variables as x
multinomModel <- multinom(Target ~ ., data=trainingSubset)

#### Running the probabilistic model to check the weights/probabilities at each level.
#### This is not used for prediction
Multinomprob <- predict(multinomModel, data = validationSubset, type ="probs")
Multinompred <- predict(multinomModel, validationSubset)
validation_labels<-validationSubset$Target

#### Confusion-matrix
cm <- table(Multinompred, validation_labels)

#### Macro f1 score - 0.165
f1_score(Multinompred, validation_labels)

#### Wald -test for significance of variables based on the beta values
z <- summary(multinomModel)$coefficients/summary(multinomModel)$standard.errors
p <- (1 - pnorm(abs(z), 0, 1)) * 2

#### Second iteration -> removing the variables that Wald-test identified as insignificant
trainingSubset <- subset( trainingSubset, select = -edjefa )
validationSubset <- subset( validationSubset, select = -edjefa )
trainingSubset <- subset( trainingSubset, select = -agesq )
validationSubset <- subset( validationSubset, select = -agesq )

#Building the model with excluding the variables
multinomModel2 <- multinom(Target ~ ., data=trainingSubset)

#Prediction using the second model
Multinompred2_2 <- predict(multinomModel2, validationSubset)
#confusion matrix
cm <- table(Multinompred2_2, validation_labels)

#### Macro f1 score - 0.1699
f1_score(Multinompred2_2, validation_labels)
```

```
################################ Random Forest ######################################
#### Random forest model
rfModel<-randomForest(as.factor(Target)~. - elimbasu5, data=training,importance=TRUE,
na.action=na.omit)

#### Variable Importance Plot to see the important predictors
varImpPlot(rfModel)

summary(rfModel)
predicted=predict(rfModel,validation,type="response")

#### Confusion Matrix
table(predicted,validation$Target)

error <- mean(validation$Target != predicted)
paste('Accuracy',round(1-error,4))

#### Macro f1 score - 0.4394
f1_score(predicted,validation$Target)

#### tuning parameters using Caret
RF = train(as.factor(Target) ~. - elimbasu5, data = training, method = "ranger", trControl =
trainControl(method = "cv"),preProcess = c("center", "scale"))

predicted1=predict(RF,validation)

#### Confusion Matrix
table(predicted1,validation$Target)

error <- mean(validation$Target != predicted1)
paste('Accuracy',round(1-error,4))

#### Macro f1 score - 0.449
f1_score(predicted1,validation$Target)
####################################################################################
```

2.Code for XGBoost

```
#Loading Necessary Packages
library(randomForest)
library(dplyr)
library(tidyverse)
library(xgboost)
library(glmnet)
library(caret)
library(nnet)


############################################ XGB #######################################

dtrain <- xgb.DMatrix(data = train_data, label= train_labels)
dtest <- xgb.DMatrix(data = test_data, label= test_labels)


#xgboost model and its parameters
xgb <- xgboost(data = dtrain,
               objective = "multi:softmax",
               booster = "gbtree",
               eval_metric = "mlogloss",
               num_class = 4,
               nthread = 4,
               eta = 0.05,
               max_depth = 8,
               min_child_weight = 6,
               gamma = 0,
               subsample = 0.7,
               colsample_bytree = 0.7,
               colsample_bylevel = 0.7,
               alpha = 0,
               lambda = 0,
               nrounds = 5000,
               print_every_n = 200,
               early_stopping_rounds = 400
)


# Plotting important features
xgb.importance(names(train_data), model = xgb) %>%
  xgb.plot.importance(top_n = 15)

#Predicting using the trained model
pred<-predict(xgb,test_data)
```

```
#Confusion matrix
cm=table(pred,test_labels)

#Function to calculate F1-Score
f1_score <- function(predicted, expected, positive.class="1") {
  predicted <- factor(as.character(predicted), levels=unique(as.character(expected)))
  expected  <- as.factor(expected)
  cm = as.matrix(table(expected, predicted))

  precision <- diag(cm) / colSums(cm)
  recall <- diag(cm) / rowSums(cm)
  f1 <-  ifelse(precision + recall == 0, 0, 2 * precision * recall / (precision + recall))

  #Assuming that F1 is zero when it's not possible compute it
  f1[is.na(f1)] <- 0

  #Binary F1 or Multi-class macro-averaged F1
  ifelse(nlevels(expected) == 2, f1[positive.class], mean(f1))
}

#Calculating F1 score for predicted model
f1_score(pred,test_labels)
povertyDataTest$Id<-0
predictions_submit = cbind(samp,pred) %>% data.frame()

names(predictions_submit) = c("Id","Target")
predictions_submit$Target = as.integer(predictions_submit$Target)

#Creating submission file
write_csv(predictions_submit, "submission.csv")

#colnames(povertyData)
#row<-nrow(povertyData)
#set.seed(12345)
#trainindex <- sample(row, row*.7, replace=FALSE)
#training <- povertyData[trainindex, ]
#validation <- povertyData[-trainindex, ]
#training
########################Random Forest model#########################
rfModel<-randomForest(as.factor(Target)~.-elimbasu5,data=training,importance=TRUE, na.action=na.omit)
#Random forst model
varImpPlot(rfModel)    #Variable Importance Plot
summary(rfModel)      #Summary of Random Forest model
predicted=predict(rfModel,validation,type="response")  #predicting our test data

table(predicted,validation$Target)    #confusion matrix

error <- mean(validation$Target != predicted)   #error rate
```

```
paste('Accuracy',round(1-error,4))    #Accuracy on our validation test


#Ensemble of random forest and XGBoost
multinomModel <- multinom(test_labels ~ pred+predicted)
#logitMod <- glm(test_labels~pred+predicted,family = )
################################################################
#Iteratively downsampling and ensembling XGBoost


dtest<- xgb.DMatrix(as.matrix(validation[,-127]))

# store predictions

prediction_list = list()

# params

params = list(
  objective = 'multi:softprob',
  eval_metric = 'merror',
  num_class = 4,
  eta = 0.1)


#Number of iterations to downsample and ensemble
n_runs = 40

for (i in 1:n_runs){
  #Down sample
  train_downsample = downSample(training[,-127], factor(training$Target))
  dtrain <- xgb.DMatrix(as.matrix(train_downsample %>% select(-Class)), label =
as.integer(train_downsample$Class)-1)
  #Cross validate to store the reults of downsample
  cv = xgb.cv(
    data = dtrain,
    nfold = 5,
    nrounds = 5000,
    verbose = FALSE,
    maximize = FALSE,
    early_stopping_rounds = 30,
    params = params,
    tree_method = "hist",
    nthread = 4
  )

  # fitting the downsampled model
  xgb.fit = xgb.train(
    data = dtrain,
    params = params,
    nrounds = round(which.min(cv$evaluation_log$test_merror_mean)*1.1,0),
    verbose = FALSE,
    maximize = FALSE,
    tree_method = "hist")

  #Prediction list for each iteration
  prediction_list[[i]] = matrix(predict(xgb.fit,dtest), ncol = 4, byrow = TRUE)
}

pred<-predict(xgb.fit,dtest)
f1_score(predictions,test_labels)
prediction = matrix(predict(xgb.fit,dtest), ncol = 4, byrow = TRUE)
predictions = apply(simplify2array(prediction_list), 1:2, mean)
predictions = apply(predictions, 1, which.max)
```

<u>Appendix B:</u>

Features	Description
Id	a unique identifier for each row.
v2a1	Monthly rent payment
hacdor	hacdor, =1 Overcrowding by bedrooms
rooms	number of all rooms in the house
hacapo	hacapo, =1 Overcrowding by rooms
v14a	v14a, =1 has bathroom in the household
refrig	refrig, =1 if the household has refrigerator
v18q	owns a tablet
v18q1	number of tablets household owns
r4h1	Males younger than 12 years of age
r4h2	Males 12 years of age and older
r4h3	Total males in the household
r4m1	Females younger than 12 years of age
r4m2	Females 12 years of age and older
r4m3	Total females in the household
r4t1	Persons younger than 12 years of age
r4t2	persons 12 years of age and older
r4t3	Total persons in the household
tamhog	size of the household
tamviv	number of persons living in the household
escolari	years of schooling
rez_esc	Years behind in school
hhsize	household size

paredblolad	paredblolad, =1 if predominant material on the outside wall is block or bri
paredzocalo	paredzocalo, "=1 if predominant material on the outside wall is socket (wo zinc or absbesto"
paredpreb	paredpreb, =1 if predominant material on the outside wall is prefabricated cement
pareddes	pareddes, =1 if predominant material on the outside wall is waste materi
paredmad	paredmad, =1 if predominant material on the outside wall is wood
paredzinc	paredzinc, =1 if predominant material on the outside wall is zink
paredfibras	paredfibras, =1 if predominant material on the outside wall is natural fibe
paredother	paredother, =1 if predominant material on the outside wall is other
pisomoscer	pisomoscer, "=1 if predominant material on the floor is mosaic, ceramic terrazo"
pisocemento	pisocemento, =1 if predominant material on the floor is cement
pisoother	pisoother, =1 if predominant material on the floor is other
pisonatur	pisonatur, =1 if predominant material on the floor is natural material
pisonotiene	pisonotiene, =1 if no floor at the household
pisomadera	pisomadera, =1 if predominant material on the floor is wood
techozinc	techozinc, =1 if predominant material on the roof is metal foil or zink
techoentrepiso	techoentrepiso, "=1 if predominant material on the roof is fiber cement, mezzanine "
techocane	techocane, =1 if predominant material on the roof is natural fibers
techootro	techootro, =1 if predominant material on the roof is other
cielorazo	cielorazo, =1 if the house has ceiling
abastaguadentro	abastaguadentro, =1 if water provision inside the dwelling
abastaguafuera	abastaguafuera, =1 if water provision outside the dwelling
abastaguano	abastaguano, =1 if no water provision
public	public, "=1 electricity from CNFL, ICE, ESPH/JASEC"

planpri	planpri, =1 electricity from private plant
noelec	noelec, =1 no electricity in the dwelling
coopele	coopele, =1 electricity from cooperative
sanitario1	sanitario1, =1 no toilet in the dwelling
sanitario2	sanitario2, =1 toilet connected to sewer or cesspool
sanitario3	sanitario3, =1 toilet connected to septic tank
sanitario5	sanitario5, =1 toilet connected to black hole or letrine
sanitario6	sanitario6, =1 toilet connected to other system
energcocinar1	energcocinar1, =1 no main source of energy used for cooking (no kitche
energcocinar2	energcocinar2, =1 main source of energy used for cooking electricity
energcocinar3	energcocinar3, =1 main source of energy used for cooking gas
energcocinar4	energcocinar4, =1 main source of energy used for cooking wood charco
elimbasu1	elimbasu1, =1 if rubbish disposal mainly by tanker truck
elimbasu2	elimbasu2, =1 if rubbish disposal mainly by botan hollow or buried
elimbasu3	elimbasu3, =1 if rubbish disposal mainly by burning
elimbasu4	elimbasu4, =1 if rubbish disposal mainly by throwing in an unoccupied space
elimbasu5	elimbasu5, "=1 if rubbish disposal mainly by throwing in river, creek or se
elimbasu6	elimbasu6, =1 if rubbish disposal mainly other
epared1	epared1, =1 if walls are bad
epared2	epared2, =1 if walls are regular
epared3	epared3, =1 if walls are good
etecho1	etecho1, =1 if roof are bad
etecho2	etecho2, =1 if roof are regular
etecho3	etecho3, =1 if roof are good
eviv1	eviv1, =1 if floor are bad

eviv2	eviv2, =1 if floor are regular
eviv3	eviv3, =1 if floor are good
dis	dis, =1 if disable person
male	male, =1 if male
female	female, =1 if female
estadocivil1	estadocivil1, =1 if less than 10 years old
estadocivil2	estadocivil2, =1 if free or coupled uunion
estadocivil3	estadocivil3, =1 if married
estadocivil4	estadocivil4, =1 if divorced
estadocivil5	estadocivil5, =1 if separated
estadocivil6	estadocivil6, =1 if widow/er
estadocivil7	estadocivil7, =1 if single
parentesco1	parentesco1, =1 if household head
parentesco2	parentesco2, =1 if spouse/partner
parentesco3	parentesco3, =1 if son/daughter
parentesco4	parentesco4, =1 if stepson/daughter
parentesco5	parentesco5, =1 if son/daughter in law
parentesco6	parentesco6, =1 if grandson/daughter
parentesco7	parentesco7, =1 if mother/father
parentesco8	parentesco8, =1 if father/mother in law
parentesco9	parentesco9, =1 if brother/sister
parentesco10	parentesco10, =1 if brother/sister in law
parentesco11	parentesco11, =1 if other family member
parentesco12	parentesco12, =1 if other non family member
idhogar	Household level identifier

hogar_nin	Number of children 0 to 19 in household
hogar_adul	Number of adults in household
hogar_mayor	# of individuals 65+ in the household
hogar_total	# of total individuals in the household
dependency	Dependency rate, calculated = (number of members of the household younger than 19 or older than 64)/(number of member of household between 19 and 64)
edjefe	years of education of male head of household, based on the interaction escolari (years of education), head of household and gender, yes=1 an no=0
edjefa	years of education of female head of household, based on the interaction escolari (years of education), head of household and gender, yes=1 an no=0
meaneduc	average years of education for adults (18+)
instlevel1	instlevel1, =1 no level of education
instlevel2	instlevel2, =1 incomplete primary
instlevel3	instlevel3, =1 complete primary
instlevel4	instlevel4, =1 incomplete academic secondary level
instlevel5	instlevel5, =1 complete academic secondary level
instlevel6	instlevel6, =1 incomplete technical secondary level
instlevel7	instlevel7, =1 complete technical secondary level
instlevel8	instlevel8, =1 undergraduate and higher education
instlevel9	instlevel9, =1 postgraduate higher education
bedrooms	number of bedrooms
overcrowding	# persons per room
tipovivi1	tipovivi1, =1 own and fully paid house
tipovivi2	tipovivi2, "=1 own, paying in installments"
tipovivi3	tipovivi3, =1 rented

tipovivi4	tipovivi4, =1 precarious
tipovivi5	tipovivi5, "=1 other(assigned, borrowed)"
computer	computer, =1 if the household has notebook or desktop computer
television	television, =1 if the household has TV
mobilephone	mobilephone, =1 if mobile phone
qmobilephone	# of mobile phones
lugar1	lugar1, =1 region Central
lugar2	lugar2, =1 region Chorotega
lugar3	lugar3, =1 region PacÃƒÂfico central
lugar4	lugar4, =1 region Brunca
lugar5	lugar5, =1 region Huetar AtlÃƒÂ¡ntica
lugar6	lugar6, =1 region Huetar Norte
area1	area1, =1 zona urbana
area2	area2, =2 zona rural
age	Age in years
SQBescolari	escolari squared
SQBage	age squared
SQBhogar_total	hogar_total squared
SQBedjefe	edjefe squared
SQBhogar_nin	hogar_nin squared
SQBovercrowding	overcrowding squared
SQBdependency	dependency squared
SQBmeaned	square of the mean years of education of adults (>=18) in the househol
agesq	Age squared

Target	the target is an ordinal variable indicating groups of income levels. 1 = extreme poverty 2 = moderate poverty 3 = vulnerable households 4 = non vulnerable households

.