NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CZ4041 Machine Learning Project

25 April 2021

Group 45	Contributions
Celine Wong Si Lin	Data cleaning Feature Engineering (General level) Modeling (Decision Tree)
Ho Xuan Rong Zane	Data cleaning Feature Engineering (Individual level) Modeling (Gradient Boosting) Feature Selection Ensemble
Koh Hui Ling	Data cleaning Feature Engineering (General level) Modeling (KNN) Feature Selection
Wang Wen	Data cleaning Feature Engineering (Household level) Modeling (Naive Bayes, Random Forest) Feature Selection

Table of Content

1. Problem Statement

Many social programs faced the difficulty of ensuring enough aid is given to the right people as typically, the poorest segments of the population are not able to provide their income and expenses records to prove that they qualify for aid. As such, Proxy Means Test (PMT) is one of the popular methods to verify income qualification by allowing agencies to use a model that considers a family's observable household attributes.
Beyond Costa Rica, many countries face the same problem of inaccurately assessing social need. Even though using PMT is an improvement, as the region's population grows and poverty declines, accuracy remains an issue. Therefore, the aim of this project is to predict the poverty level accurately given the dataset of Costa Rican household characteristics.

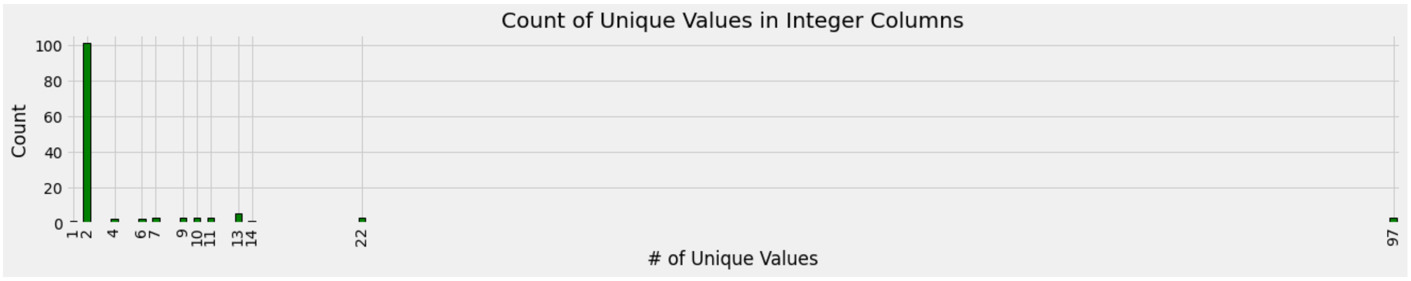
2. Challenges

The competition's main challenge is the problem of imbalance class which can lead to our machine learning models failing to predict those in the minority groups due to the lack of data. In addition, since the scoring of the competition is based only on the predictions for heads of household, we need to find ways to aggregate individual household members' data to household level.

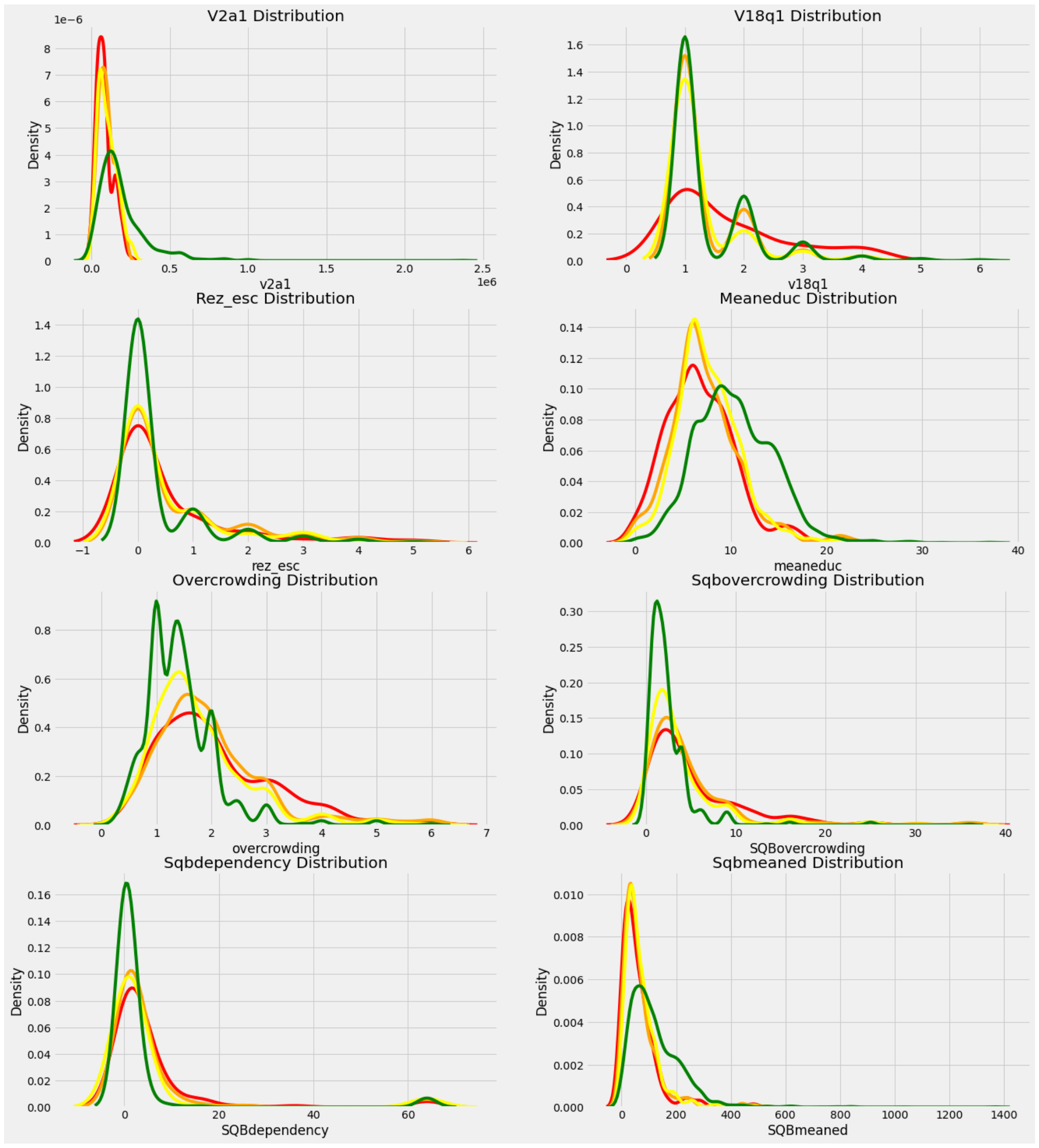
3. Proposed Solution

3.1 Data Cleaning and Exploration

Integer Columns
The bar chart below shows that the dataset contains a high number of boolean data. In most cases, this boolean information is already on a household level.



Float Columns
In the following graphs, each color represents a different value of the Target and we can observe that the variable distribution varies significantly depending on the household poverty level.

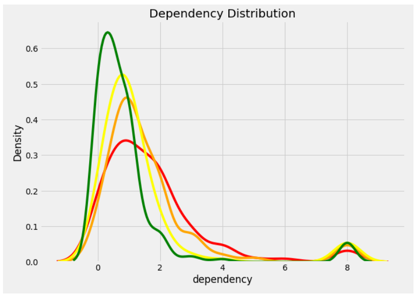


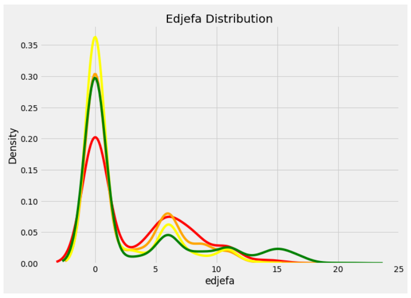
<u>Object Columns</u>

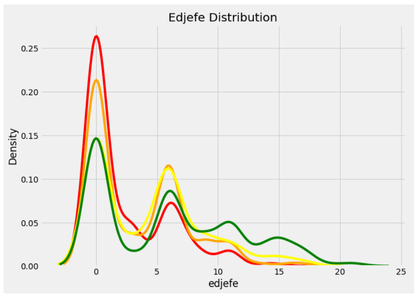
Id and idhogar have object types which make sense because these are identifying variables. However, the variables below are object types, having a mix of strings and numbers which need to be addressed.

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 of escolari (years of education), head of household and gender
edjefa	years of education of female head of household, based on the interaction of escolari (years of education), head of household and gender

For these three variables, ‘yes’ equals 1 and ‘no’ equals 0. We can correct the variables using mapping and convert them to floats.

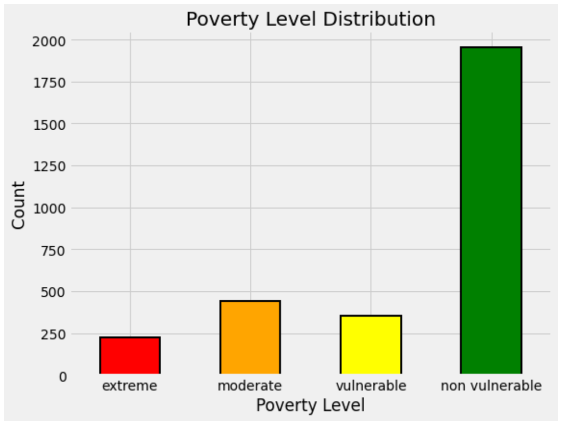






Exploring Label Distribution

To explore the label distribution, we only look at instances where parentesco1 == 1 because this is the head of household, the correct label for each household.



The chart shows that there is an imbalance class problem. With fewer examples for some classes, the machine learning models can struggle to predict the minority groups.

Addressing Wrong Labels

We found that some labels are incorrect as individuals in the same household have a different poverty level. Thus, we need to correct it.

Firstly, we check if there is only one unique Target value for each household and found that there are 85 households where family members do not have the same target label.

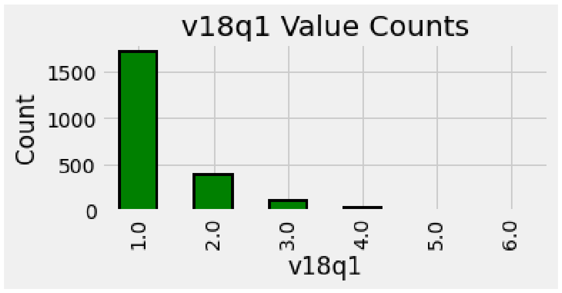
Based on the competition discussion, the organizers mentioned that the correct label is the head of household. As such, we have corrected the labels for households with members having different poverty levels by assigning the true label for all members.

Outlier

There is one outlier in the rez_esc column. According to the competition discussions, the maximum value for this variable is 5. As a result, all values greater than 5 should be set to 5.

Missing Variables

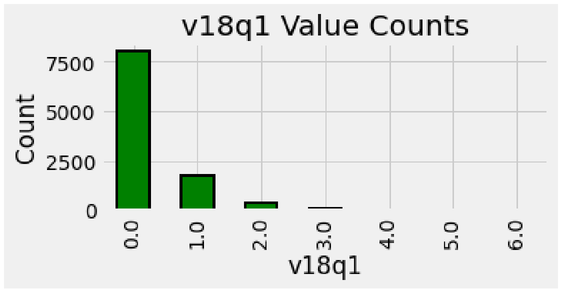
As v18q1 is a household variable, when looking into the value counts of it, we'll only select the rows for the head of household.



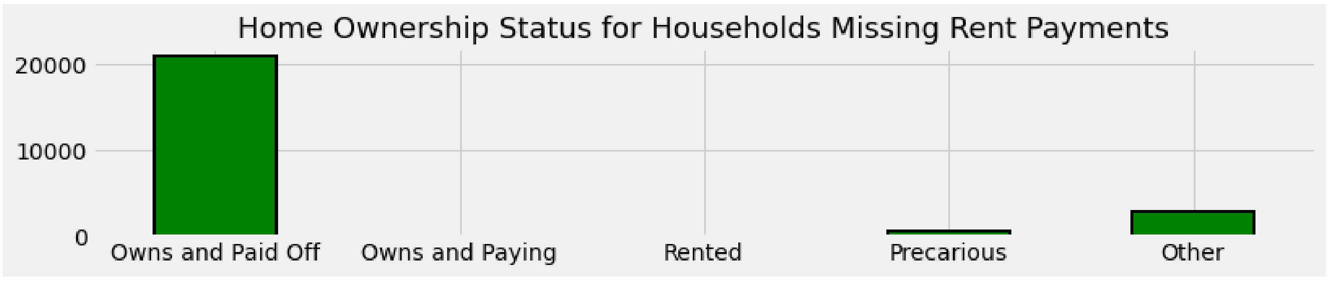
According to the current statistics, the majority of households own one tablet. However, we must also consider the data that is missing. It is possible that families with a nan in this category simply do not own a tablet. According to the data definitions, v18q shows whether or not a family owns a tablet.

We are able to groupby the value of v18q, where 1 means owns a tablet and 0 means does not own a tablet, and then calculates the number of null values for v18q1. This will tell us whether the null values represent that the family does not own a tablet.

Since every family that has nan for v18q1 does not own a tablet. Therefore, we can fill in this missing value with zero.



The column v2a1 shows the monthly rent payment which is the next missing column. We will also look at the distribution of tipovivi_, the columns showing the ownership or renting status of the home. For the following plot, we show the ownership status of those homes with a nan for the monthly rent payment.



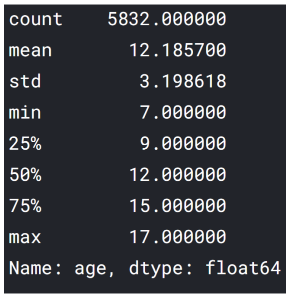
The meaning of the home ownership variables are:

tipovivi1	=1 own and fully paid house
tipovivi2	=1 own, paying in installments
tipovivi3	=1 rented
tipovivi4	=1 precarious
tipovivi5	=1 other(assigned, borrowed)

Thus, we can see that the households that do not have monthly rent payment generally own their own home.

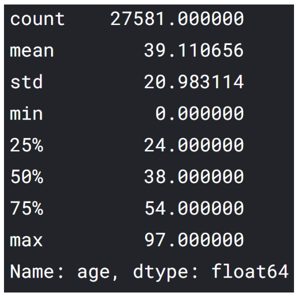
For the houses that are owned and have a missing monthly rent payment, we can set the value of the rent payment to zero. For the other houses, we can leave the missing values to be imputed but we will add a boolean flag column indicating that these households had missing values.

The column rez_esc shows the years behind in school. For the families with a null value, it is possible that they currently do not have any children in school. We are able to test this out by finding the ages of those who have a missing value in this column and the ages of those who do not have a missing value.



```
count    5832.000000
mean       12.185700
std         3.198618
min         7.000000
25%         9.000000
50%        12.000000
75%        15.000000
max        17.000000
Name: age, dtype: float64
```

According to this, the oldest age with a missing value is 17. We should probably assume that someone older than this is actually not in school. Next, we will look into the ages of those who have a missing value.



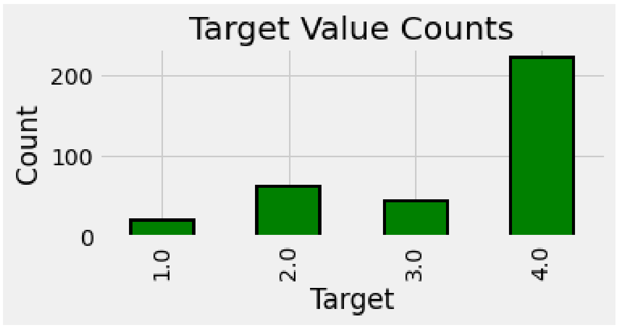
```
count    27581.000000
mean        39.110656
std         20.983114
min          0.000000
25%         24.000000
50%         38.000000
75%         54.000000
max         97.000000
Name: age, dtype: float64
```

After reading the competition discussions, we discover that this variable is only specified for people aged 7 to 19. Anyone younger or older than this age range is likely to be no years behind, so the value should be set to 0. Therefore, we have set this variable to zero if the individual is over 19 and has a missing value, or if they are younger than 7 and have a missing value. We will leave the value to be imputed for anyone else and add a boolean flag.

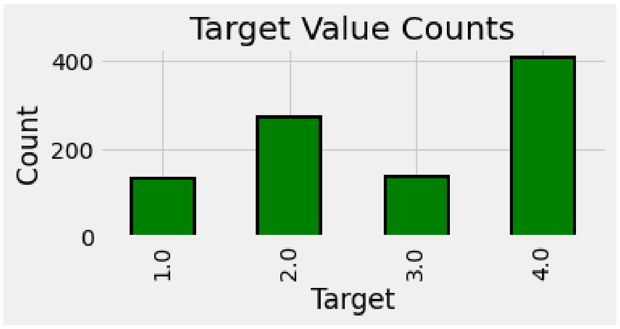
Target Label Distribution of missing value

As a final step with the missing values, we can plot the distribution of target for the case where either of these values are missing.

Missing values of rez_esc



Missing values of v2a1



For rez_esc, the distribution seems to match the label distribution for all data whereas the distribution for v2a1 could be an indicator of more poverty given to the higher prevalence of 2 (moderate poverty). This illustrates that sometimes the missing information is just as important as the information given.

Impute missing value

Imputation is the process of filling in the remaining missing values in each column. There are many forms of imputation that are commonly used, but we have chosen to fill in missing values with the column's median since it is one of the simplest and most effective methods.

3.2 Feature Engineering

By performing Feature Engineering, we transformed existing data into more meaningful input data that is compatible with the selected training models and aims to improve the model's performance. In this project, our team performed 3 levels of feature engineering: General Level, Household Level, and Individual Level.

To aid in the feature engineering operations later, we categorised the variables into four categories and further differentiate them according to their data types as follows:

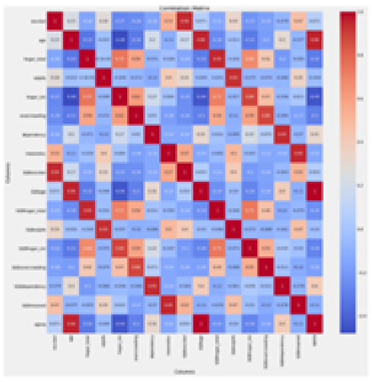
Category	Purpose	Array
Id Variables	To identify the data in the dataset	**id_** = ['Id', 'idhogar', 'Target']
Individual Variables	Variables consist characteristic of boolean and ordered discrete that are used to describe data in terms of Individual related.	**ind_bool** = ['v18q', 'dis', 'male', 'female', 'estadocivil1', 'estadocivil2', 'estadocivil3', 'estadocivil4', 'estadocivil5', 'estadocivil6', 'estadocivil7', 'parentesco1', 'parentesco2', 'parentesco3', 'parentesco4', 'parentesco5', 'parentesco6', 'parentesco7', 'parentesco8', 'parentesco9', 'parentesco10', 'parentesco11', 'parentesco12', 'instlevel1', 'instlevel2', 'instlevel3', 'instlevel4', 'instlevel5', 'instlevel6', 'instlevel7', 'instlevel8', 'instlevel9', 'mobilephone', 'rez_esc-missing'] **ind_ordered** = ['rez_esc', 'escolari', 'age']
Household Variables	Variables consist of boolean, ordered discrete and continuous value that used to describe data in terms of Household related.	**hh_bool** = ['hacdor', 'hacapo', 'v14a', 'refrig', 'paredblolad', 'paredzocalo', 'paredpreb','pisocemento', 'pareddes', 'paredmad', 'paredzinc', 'paredfibras', 'paredother', 'pisomoscer', 'pisoother', 'pisonatur', 'pisonotiene', 'pisomadera', 'techozinc', 'techoentrepiso', 'techocane', 'techootro', 'cielorazo', 'abastaguadentro', 'abastaguafuera', 'abastaguano', 'public', 'planpri', 'noelec', 'coopele', 'sanitario1', 'sanitario2', 'sanitario3', 'sanitario5', 'sanitario6', 'energcocinar1', 'energcocinar2', 'energcocinar3', 'energcocinar4', 'elimbasu1', 'elimbasu2', 'elimbasu3', 'elimbasu4', 'elimbasu5', 'elimbasu6', 'epared1', 'epared2', 'epared3', 'etecho1', 'etecho2', 'etecho3', 'eviv1', 'eviv2', 'eviv3', 'tipovivi1', 'tipovivi2', 'tipovivi3', 'tipovivi4', 'tipovivi5', 'computer', 'television', 'lugar1', 'lugar2',

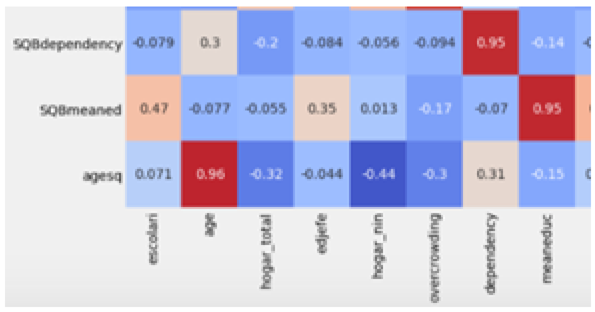
		'lugar3', 'lugar4', 'lugar5', 'lugar6', 'area1', 'area2', 'v2a1-missing'] **hh_ordered** = ['rooms', 'r4h1', 'r4h2', 'r4h3', 'r4m1','r4m2','r4m3', 'r4t1', 'r4t2', 'r4t3', 'v18q1', 'tamhog','tamviv','hhsize','hogar_nin', 'hogar_adul','hogar_mayor','hogar_total', 'bedrooms', 'qmobilephone'] **hh_cont** = ['v2a1', 'dependency', 'edjefe', 'edjefa', 'meaneduc', 'overcrowding']
Squared Variables	To derive as new variable by squaring variables in the dataset	**squared =** ['SQBescolari', 'SQBage', 'SQBhogar_total', 'SQBedjefe', 'SQBhogar_nin', 'SQBovercrowding', 'SQBdependency', 'SQBmeaned', 'agesq']

3.2.1 General Level

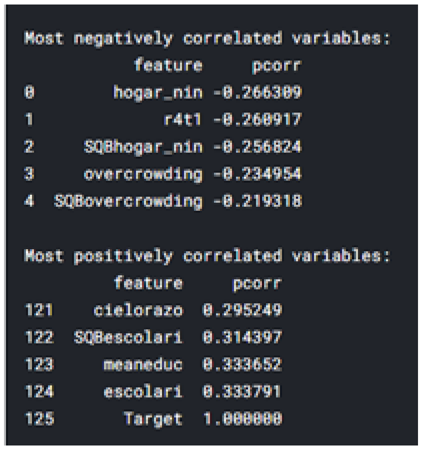
In the general level, we dropped some redundant variables, explored the features to have a good understanding of the dataset and identified the focus area in constructing features in the household level and individual level.

The Correlation Matrix below shows that the squared variables*: SQBescolari, SQBage, SQBhogar_total, SQBedjefe, SQBhogar_nin, SQBovercrowding, SQBdependency, SQBmeaned and agesq* are highly correlated (>= 0.95) with their original un-squared variables. Thus, we dropped them to reduce the dimension of the table.



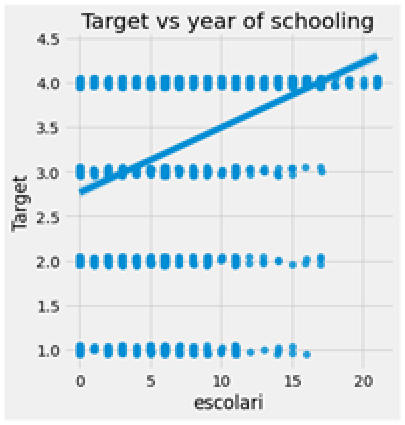


After making some reduction in table dimensions and exploration, we have concluded the top 5 most positively and negatively correlated variables as shown in the output below. We noticed that the positively correlated features tend to describe the education situation in the household. From the plot shown below, the household with a lesser year of schooling tends to be poorer.

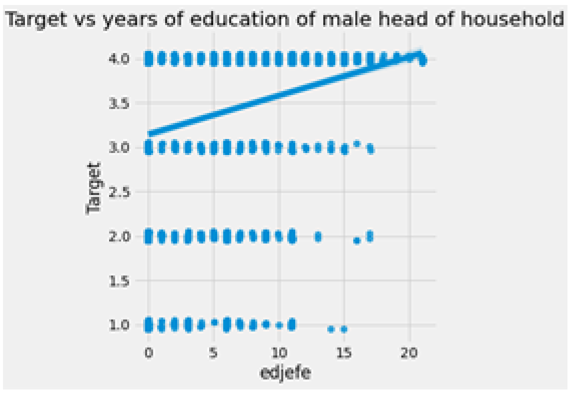


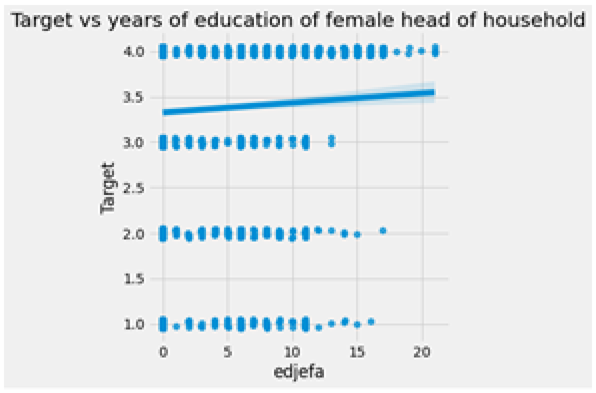
```
Most negatively correlated variables:
            feature      pcorr
0         hogar_nin -0.266309
1              r4t1 -0.260917
2      SQBhogar_nin -0.256824
3      overcrowding -0.234954
4   SQBovercrowding -0.219318

Most positively correlated variables:
          feature      pcorr
121     cielorazo  0.295249
122  SQBescolari  0.314397
123       meaneduc  0.333652
124       escolari  0.333791
125        Target  1.000000
```



By exploring the years of education of the male and female head of the household, the plots below show that the years of education for male head have a higher positive correlation with target than female head.





3.2.2 Household Level

For household level of feature engineering, we subset the dataset based on the head of the household where *parentecol1==1* as it is used to identify the labels used for training. Then, we created a new dataframe, *hh* to hold all the household variables by adding arrays *id_*, *hh_bool*, *hh_cont*, and *hh_ordered*.

As the prediction is conduct based on the household variables, we will remove some redundant variables and keep most of the household variables to use as the features. Based on the latest version of the dataset, we identified that some household variables are highly correlated by finding the index of the column with a correlation value greater than 0.95. They are *coopele, area2, tamhog, hhsize, hogar_total.* Furthermore, we subset the correlation matrix which helps us to identify more pairs such as *tamhog, hogar_total, r4t3, hhsize and tamviv*. Hence, we removed the redundant household variables and kept the variable *hhsize* to identify household size and *tamviv* for the number of people not from the same household but living in the same house.

Construction of New Features

By using simple calculation (-, +, *, /), we derived new features to denote the household population information based on the household size. Higher child and elderly ratio in a household will definitely lead to higher dependency rate and more financial stress on the head of household.

New Feature	**Derivation**
adult	No of adult in household - # individuals 65+ years old in the household
dependency_count	Number of children 0 to 19 in household + # of individuals 65+ in the household
dependency	dependency count / adult <65 years old
child_percent	Number of children 0 to 19 in household / household size
elder_percent	# of individuals 65+ in the household / household size
adult_percent	Number of adults in household / household size
males_younger_12_years_in_household_size	Males younger than 12 years of age / household size
males_older_12_years_in_household_size	Males 12 years of age and older / household size
males_in_household_size	Total males in the household/household size
female_younger_12_years_in_household_size	females younger than 12 years of age / household size
females_older_12_years_in_household_size	females 12 years of age and older / household size
females_in_household_size	Total females in the household / household size
persons_younger_12_years_in_household_size	Persons younger than 12 years of age / household size
persons_older_12_years_in	Persons older than 12 years of age / household size

_household_size	
overcrowding _room_and_bedroom	Overcrowding by bedrooms =1 + Overcrowding by rooms =1
age_12_19	Number of children 0 to 19 in household - persons younger than 12 years of age

Construction of Per Capita Features

By using simple calculation (-, +, *, /), we derived new features in some essential factors such as housing and electronic gadgets that potentially represent the poverty level of a household based on per capita. By using per capita instead of actual numbers, it helps to remove biases due to different household size.

New Feature	**Derivation**
phones-per-capita	# of mobile phones/# of person living in the household
tablets-per-capita	# of tablets household owns/# of person living in the household
rooms-per-capita	# of all rooms in the house/# of person living in the household
rent-per-capita	monthly rent payment/# of person living in the household

Combination of Relevant Features Per Household

In this section, we generate more household features by multiplying each feature to form a new column formatted as [new_{}_x_{}](e.g. [*new_lugar1_x_meaneduc*]). By mixing and grouping relevant features in the same topic together, it is efficient to understand what factor contributes to the poverty level of a household.

New Feature	**Derivation**
Electronic Feature	Number of electrical devices per household by combining computer, mobilephone, television, refrige and tablet
	Equation: Computer*qmobilephone*television*refrige*tablet
Living Environment	Living condition per household by combining the conditions and materials used in walls, roof, and floor.

Feature	**Equation:** wall condition*roof condition*floor condition and wall material*roof material*floor material
Water Feature	Water accessibility per household by combining the water provision and toilet infrastructure
	Equation: water provision*toilet
Hygiene Feature	Hygiene issues per household by combing the toilet condition and rubbish disposal methods used
	Equation: Toilet*rubbish
Education Feature	Average education per household from different regions
	Equation: region*average year of schooling

In addition, for newly generated columns with only one value, we have dropped it to reduce the dimensions. Thus, there are **396 features** in total after feature engineering at the household level.

3.2.3 Individual Level

For feature engineering at the individual level, we also subset the individual variables by using a dataframe, *ind* to hold all the variables from the arrays *id_, ind_bool,* and *ind_ordered*. The individual variables with a correlated value greater than 0.95 are dropped as they are highly correlated to the target. Lastly, encoded the 9 instlevel variables into an ordinal column by finding the non-zero columns. The inslevel indicates the amount of education an individual has from instlevel1: no level of education(lowest) to instlevel9: postgraduate education (highest) which makes sense to replace with an ordinal variable.

Moreover, there are more features constructed based on the individual variables using simple calculation (-, +,*,/) as shown in the table below.

New Feature	**Derivation**
escolari/age	individual years of schooling/age in years
inst/age	individual education level/age in years
tech	individual owns a tablet + individual owns a mobile phone

Aggregate individual data to household level

To incorporate individual data to household level, we have aggregated each individual variable by grouping using their household id and used various aggregation functions (min, max, sum, count, standard deviation', range) on each individual variable. Finally, we dropped all highly correlated columns with correlated value greater than 0.95 and merged with the household level features. The final feature set has **491 features**.

3.3 Modelling

To create baseline models, we experimented and fitted various algorithms that are suitable for supervised multi-class classification problems such as Decision Tree, Naive Bayes, K-Nearest Neighbour (KNN), Random Forest and Gradient Boosting.

3.3.1 Decision Tree

We first fitted a tree using all available features but found that by limiting the input features to *meaneduc* and *cielorazo* (the top two highly correlated features to the target), we are able to obtain a higher accuracy. We also set max_depth=3 to prevent overfitting.

3.3.2 Naive Bayes

There are three types of algorithms: Binomial, Multinomial and Gaussian. In this project, we have chosen Gaussian Algorithm as it looks at the distribution count of the selected features and determines where the cut-off point to classify the household into the 4 different poverty levels. However, the Binomial Algorithm only classifies if output is 0 or 1 and Multinomial Algorithm is mainly used in text classification by calculating the frequency of word appearance. Hence, Gaussian Algorithm is most suitable to use in our project. We also are not adding the parameter as it makes strong independence assumptions.

3.3.3 KNN

For KNN, we have tried various values of K and choose K=12 as it has the highest accuracy.

3.3.4 Random Forest

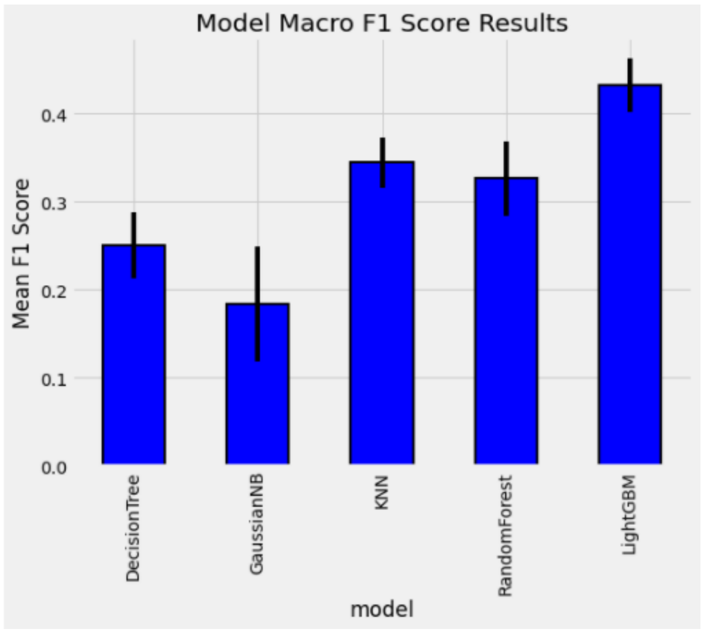
Random Forest is a more advanced model that combines multiple decision trees to get a more accurate and stable prediction. We set the n_estimators to 100 as the number of the trees we want to build before taking the highest voting. Although the higher n_estimator leads to better performance but it is constrained by the limitation in time,memory and processor. We have no restriction in the number of processors allowed to use by setting the n_jobs=-1 as we do not want to waste resources by over-allocating processors to the task. Lastly, we set random_state=10 to ensure that the output of each run will be the same.

3.3.5 Gradient Boosting

Gradient Boosting is also a technique that combines decision trees like the Random Forest, but the combination process starts at the beginning instead of the end. Hence, Gradient Boosting produces a prediction model in the form of an ensemble of weak prediction models.
XGBoost (Extreme Gradient Boosting) has been the de-facto choice in many data science competitions simply because it is extremely powerful. However, Light Gradient Boosting Machine (LGBM) improves on XGBoost as it outperformed in training speed while attaining comparable performance. Therefore, we decided to use only LGBM under the Gradient Boosting type.
For the LGBM parameters, to deal with imbalance class problems, we have set the class_weight='balanced' to assign the class weight inversely proportional to their frequencies. We used early_stopping_rounds=400 to stop training once the validation score does not improve; and colsample_bytree=0.93 to randomly select a subset 93% of features for each iteration which helps to prevent overfitting and reduce training time. To achieve better accuracy, we have decreased learning rate to 0.03.

3.3.6 Model Performance Comparison

To compare the performance of these models, we used the 5-Fold Stratified Cross Validation. We chose Stratified Cross Validation as it helps keep the same proportion of different classes in each fold which is useful to counter imbalance data problems. We have set the cross validation scorer to be Macro F1 Score, the same used by the project organiser. The performance of each model is shown below:



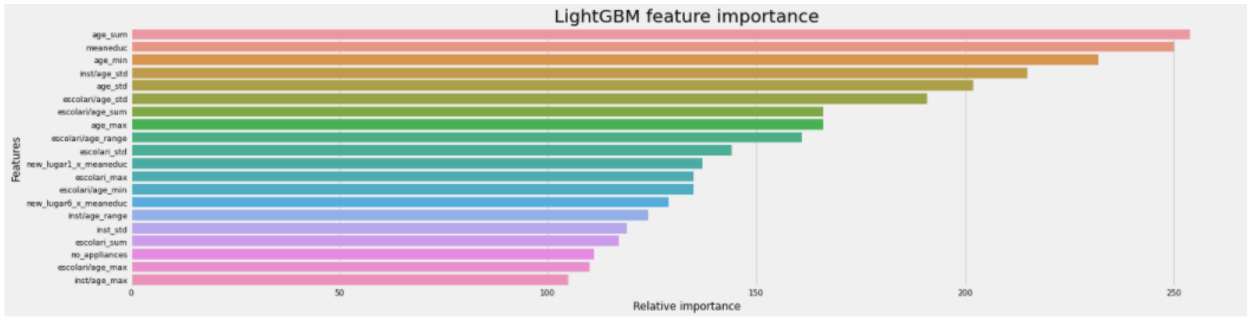
Based on the baseline performance above, we decided to rule out Decision Tree and Naives Bayes and proceeded to experiment with KNN, Random Forest and LGBM to tune the parameters and further improve their performance.

3.4 Experiments

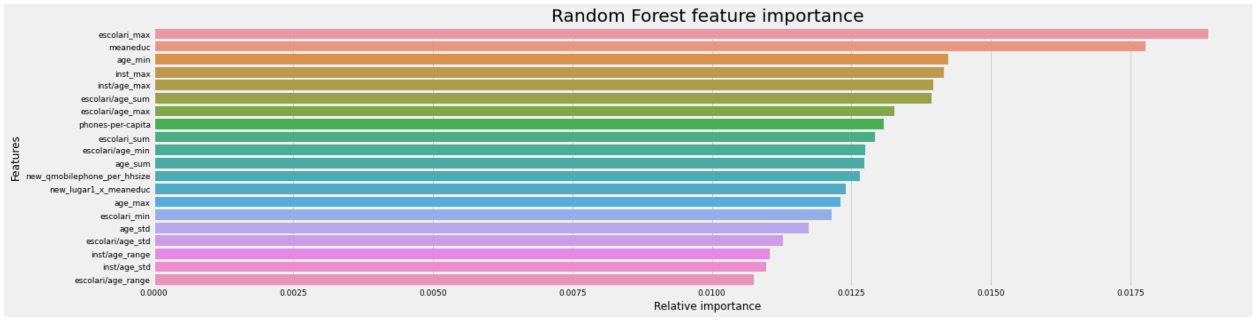
3.4.1 Feature Importance

After the modelling step above, we are able to compare the features important of LGBM and Random Forest but not for KNN as it does not expose the "coef_" or "feature_importances" attribute.

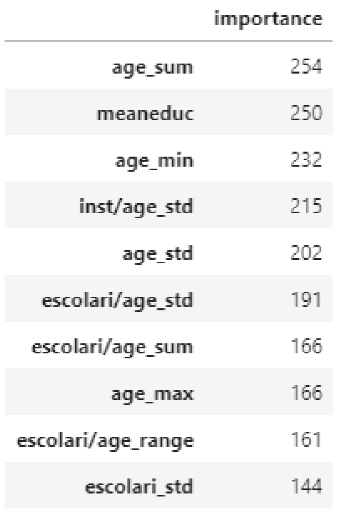
Feature Importance generated from LGBM model



Feature Importance generated from Random Forest model

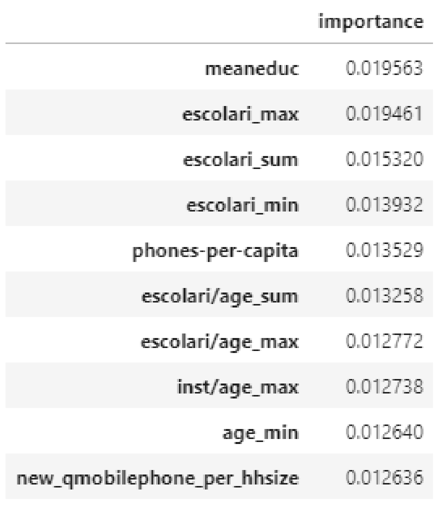


LGBM Feature Ranking



	importance
age_sum	254
meaneduc	250
age_min	232
inst/age_std	215
age_std	202
escolari/age_std	191
escolari/age_sum	166
age_max	166
escolari/age_range	161
escolari_std	144

Random Forest Feature Ranking



	importance
meaneduc	0.019563
escolari_max	0.019461
escolari_sum	0.015320
escolari_min	0.013932
phones-per-capita	0.013529
escolari/age_sum	0.013258
escolari/age_max	0.012772
inst/age_max	0.012738
age_min	0.012640
new_qmobilephone_per_hhsize	0.012636

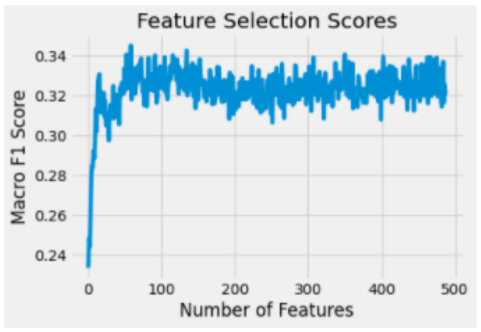
We can observe from the result that both models have ranked the features with slight differences. Some features that have high importance in both models include: *meaneduc*, *age_min*, *escolari* and *age_sum*.

3.4.2 Feature Selection

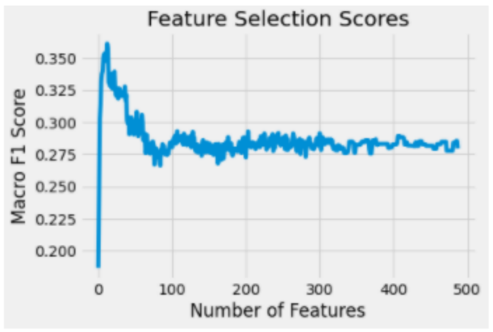
After the feature engineering process, our final feature set is 491 which is a huge number of features. Therefore, we have decided to experiment with feature selection to select a subset of features that contribute most to the prediction and to discover the relationship between the number of features used and the accuracy of our model.

A popular and effective feature selection algorithm is Recursive feature elimination (RFE) which fits a model and recursively removes the weakest feature until the specified number of features is reached. RFE also attempts to eliminate dependencies and collinearity that may exist in the model. We have decided to include cross-validation with feature selection to mitigate overfitting, thus we have used Recursive Feature Elimination and Cross-Validation Selection (RFCEV).

RFECV on Random Forest

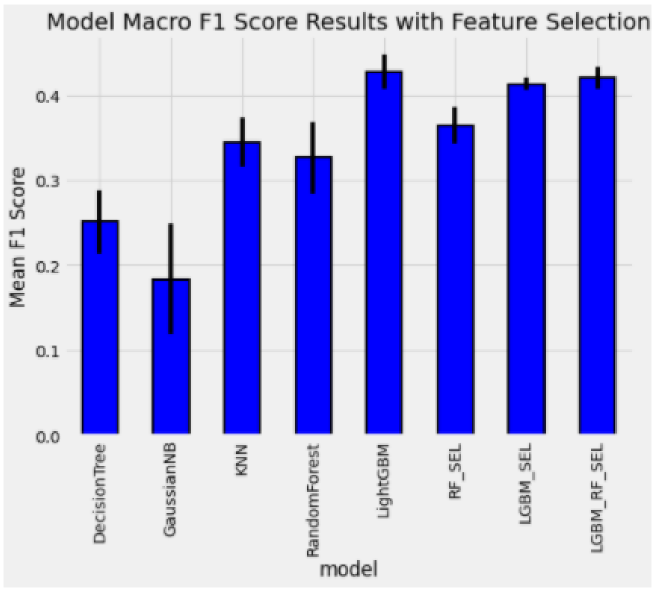


RFECV on LGBM



The chart above shows the results of performing RFECV on Random Forest and LGBM models used in the modelling step. For Random Forest, the macro F1 score peaked at 59 features with an oscillating trend afterwards and generally decreased slightly as more features were used.
For LGBM, the macro F1 Score peaked at 13 features and decreased dramatically as more features are used.

Therefore, in our experiment, we retrained the Random Forest model using the top 59 features (RF_SEL), LGBM using the top 13 features (LGBM_SEL) and LGBM using the top 59 features (LGBM_RF_SEL).



From this experiment, we observed that the score for Random Forest with feature selection improved significantly. However, the score for LGBM with feature selection score remained comparable as before.

3.4.3 Ensemble of different algorithm

In an attempt to improve accuracy, we further experimented with ensembling different algorithms. We decided to use a voting ensemble which is an ensemble machine learning model that combines the predictions from multiple other models to achieve better results.

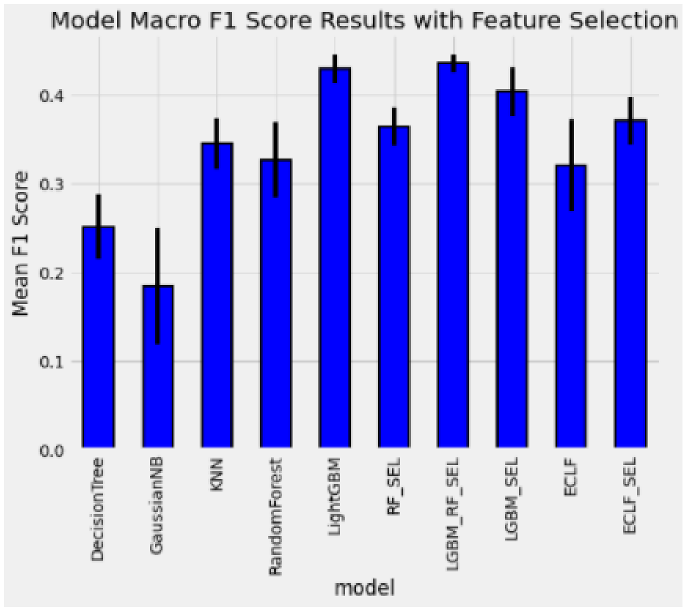
We decided to ensemble the KNN, RF and LGBM models that we used previously in the modelling and feature selection step as they are the models with the highest score. In addition, we have observed that their predictions are merely moderately positively correlated which indicate they are suitable candidates to be ensembled.

	Target_LGBM	Target_RF	Target_KNN
Target_LGBM	1.000000	0.671520	0.565287
Target_RF	0.671520	1.000000	0.542704
Target_KNN	0.565287	0.542704	1.000000

To produce the prediction, we have used soft voting which involves summing the predicted probabilities for each class label and use the class label with the largest probability as the prediction.

The two ensemble classifier experimented are:

- ECLF: ensemble of KNN, RF and LGBM using whole feature set
- ECLF_SELL: ensemble of KNN, RF and LGBM using top 59 features.



```
          model   cv_mean    cv_std
0  DecisionTree  0.251090  0.037367
1    GaussianNB  0.183490  0.065128
2           KNN  0.344741  0.028629
3  RandomForest  0.326392  0.042558
4      LightGBM  0.429687  0.015900
5        RF_SEL  0.364305  0.021577
6   LGBM_RF_SEL  0.435754  0.009537
7      LGBM_SEL  0.403684  0.027366
8          ECLF  0.320471  0.051235
9      ECLF_SEL  0.370854  0.026907
```

The score of the ECLF did poorer with higher standard deviation while ECLF_SEL attained a comparable score with RF_SEL but is still worse off than using LGBM alone.

3.5 Final Chosen Model

Finally, after modelling and experimenting the various techniques to achieve better accuracy, the model used for the competition is **LGBM_RF_SEL** (LGBM with the top 59 features) as it has the highest Macro F1 mean Score and lowest standard deviation.

4. Conclusion

1. Importance of Feature Engineering

   Throughout this project, we learned that the crucial step in having good model performance is feature engineering. It is essential to have quality features compared to quantity. By constructing the features, we can explore those variables highly positively correlated to the target and filter out the redundant variables to reduce the dimension. The input feature to the model will determine the performance of the models.

2. Parameters contribute to Model's performance

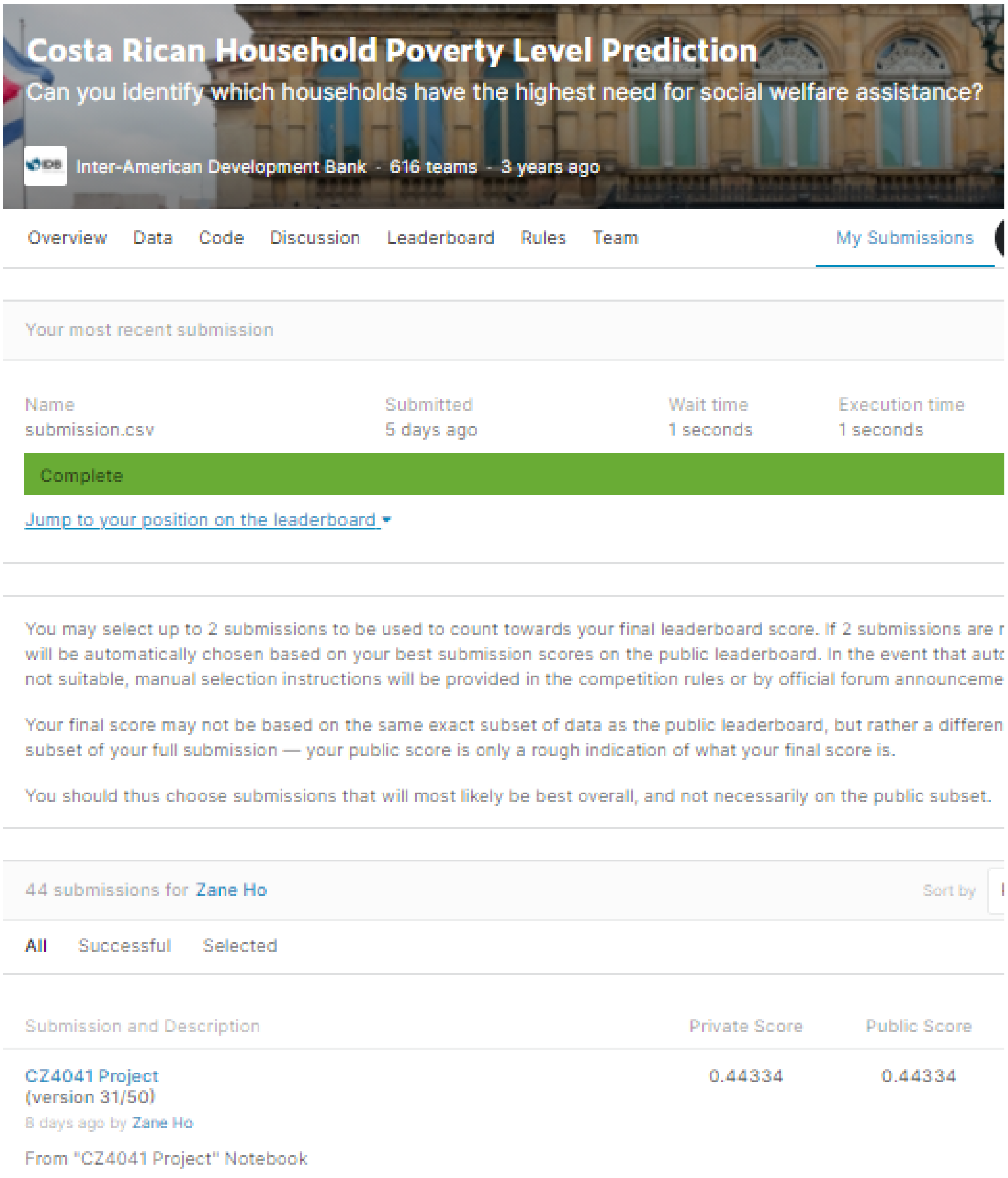
   In some models we used in the project, we could have better performance by testing different values for parameters used in the models. For example, in the Random Forest modelling, the F1 score could be better if we increase the n_estimators (number of trees to build) to a higher value. The higher number of trees built, the better performance the model will get. However, it is often trade-off by the memory and time consumption.

3. Memory and Time Consumption in Project

   We also learned that running complex model algorithms on a large dataset required a sufficient amount of CPU memory as it takes up a lot of resources and time running in the Kaggle platform. It will be better to use an alternative platform such as Google Colab which has a longer execution time and faster running time.

5. Kaggle Leaderboard Screenshot

Best public & private score: 0.44334
Public ranking: # 22 out of 616
Private ranking: # 21 out of 616



Costa Rican Household Poverty Level Prediction
Can you identify which households have the highest need for social welfare assistance?
Inter-American Development Bank · 616 teams · 3 years ago

Overview Data Code Discussion Leaderboard Rules Team My Submissions

Your most recent submission

Name	Submitted	Wait time	Execution time
submission.csv	5 days ago	1 seconds	1 seconds

Complete

Jump to your position on the leaderboard

You may select up to 2 submissions to be used to count towards your final leaderboard score. If 2 submissions are r
will be automatically chosen based on your best submission scores on the public leaderboard. In the event that autc
not suitable, manual selection instructions will be provided in the competition rules or by official forum announceme

Your final score may not be based on the same exact subset of data as the public leaderboard, but rather a differen
subset of your full submission — your public score is only a rough indication of what your final score is.

You should thus choose submissions that will most likely be best overall, and not necessarily on the public subset.

44 submissions for Zane Ho — Sort by

All Successful Selected

Submission and Description	Private Score	Public Score
CZ4041 Project (version 31/50) 8 days ago by Zane Ho From "CZ4041 Project" Notebook	0.44334	0.44334