Udacity Machine Learning Nanodegree - Capstone Project

Ruslan Kozhuharov

June 9, 2018

Contents

1 Definition

1.1 Project Overview

The Kiva organization is a charitable entity that funds underprivileged people around the world through small loans. Such loans are funded on Kivas online platform by one or several donors. The loan amount is then disbursed by Kiva associates to financially excluded individuals. Kiva itself does not collect rent on those loans, but Kiva associates could in order to cover their operating costs. Once a loan is repaid, the individual donors could choose another cause and continue their charitable activity. This helps poor communities by:

- Fostering economic activity
- Providing access to funds for financially excluded individuals
- Encouraging entrepreneurial initiative on the part of the borrower

In order to improve their reach, however, Kiva needs a good assessment of the poverty levels in their areas of operation. To do that, Kiva has requested in a Kaggle challenge that a poverty score is created based on other data and that score is combined with a data set for Kivas loans from the last 2 years.

The prior research in this area consists of two main bodies: the Human Poverty Index, developed in 1997 and the Multidimensional Poverty Index, introduced in 2010. The Human Poverty Index (HPI) focuses on the following metrics for defeloping countries:

- Mortality
- Illiteracy
- Living conditions (access to water and access to nutrition)

For the so called 'high income OECD countries', the HPI uses the following metrics:

- Mortality
- Illiteracy
- Income
- Unemployment

The HPI is statistically-derived result that uses the formula:

$$HPI = \left[\frac{\sum_{i=1}^{n} P_i^{\alpha}}{n}\right]^{\frac{1}{\alpha}} \tag{1}$$

That is, the HPI is not predictive, but descriptive. In 2010, the HPI was replaced by the Multidimensional Poverty Index (MPI). As with the HPI, the MPI is a descriptive index. The MPI is based on more aspects of poverty, namely:

- Child mortality
- Nutrition
- Years of education
- School attendance
- Energy and electricity

- Water
- Sanitation
- Accomodation
- Assets

The indicators above have different weights in the index and their sumproduct is weighted by the so-defined poverty's intensity.

We will attempt to develop a predictive model that uses various factors mentioned above as predictors of lack of income. In this respect our model is more singular than the HPI and MPI, but on the other hand could use the indicators present in both indices as predictors.

1.2 Problem Statement

Based on the definition of the Kiva challenge laid out in the Project Overview, the problems we are going to solve are the following:

1. Defining the mathematical expression of poverty
2. Selecting the necessary regional data sets
3. Transforming the data set from step 2 to a form usable for modeling
4. Building the poverty prediction model
5. Joining the predicted poverty to all records of the Kiva loans data set

1.2.1 Defining Poverty

As mentioned above, poverty is a subjective notion that we need to convert to a mathematical measure. In defining that measure we need to take into account the following factors:

- The measure should be applicable on country as well as on global level.
- The measure should be on a scale between 0 to 1 so that it represents a percentage of poverty. This will allow Kiva to report on poverty-affected areas with regards to the measures maximum.
- The measure should be scalable if corresponding data from more countries is added.

With regards to these points, we will define the poverty score as: 1 - predicted normalized total household income. That is, poverty score of 0.8 corresponds to predicted normalized total household income of 0.2. Thus our prediction will always be on scale from 0 to 1, will be applicable to country and global levels and will be scalable as more country profiles are added (the normalized household income will always be from 0 to 1).

1.2.2 Selecting Data Sets

There are two main problems that need to be resolved when selecting data sets useful for the modeling task:

- Data with the widest possible coverage (World Bank, UN statistics, CIA World Factbook) is averaged out for each country and does not present a nuanced enough picture of the poverty levels. That is, in countries with high GINI coefficient, the average income levels do not provide a sufficient information about areas with a significant deviation from the average (poverty affected areas).

- Local country data is presented in different formats. Every country measures different macroeconomic indicators (and sometimes measures the same indicators in different ways). This makes local data, that could eventually be joined together on global level, unreliable. Example: some countries may consider 'unemployment' as the number of working age individuals with no regular employment contract as unemployed. Other countries may consider 'unemployment' as individuals who do not have a regular employment contract AND are looking for employment (i.e. individuals who are not actively looking for jobs are not counted).

To address these issues, we will select the country where Kiva grants the highest number of loans. We will focus exclusively on that country and find a detailed dataset, preferably containing raw data. We will then build an adequate model for the selected country and reduce the number of variables to a few, easily obtainable for other countries. This will make our model, although localized, scalable.

The country receiving the most Kiva loans at this point is the Philippines. Therefore, we will use data from the Philippines Family Income and Expenditure triennial survey (FIES). The data contains information about families incomes, expenses, and living conditions (with detailed information about accommodation types, running water, electricity, communications devices, family size, education, etc.). The data set is published on Kaggle and is produced by the Philippines Statistics Authority (PSA).

This data set will be used to derive a model that can predict poverty out of the available features. The poverty score will then be grouped by region and household head gender. We will also derive the administrative regions from the Kiva data set and join the calculated poverty score by region and gender of the borrower.

1.2.3 Transforming The Data

Since the data set we will utilize will contain raw data, we will need to transform it to numerical or one hot encoded features. We will perform this work separately in a data transformation phase.

1.2.4 Building The Poverty Prediction Model

Once we have transformed the FIES data, we will need to find eventual correlations, remove irrelevant fields and prepare for modeling. We will then proceed and try the effectiveness of several out of the box models (we will refer to them as candidate models) against three dummy predictors.

The dummy predictors will always predict poverty scores of 0, 0.5 and 1, respectively. We will then compare the performance of the candidate models against the dummy predictors based on the evaluation metrics defined in the next chapter. Finally, we will select the best performing model and improve on it.

The improvement will consist of selecting the features with highest predictive potential and discarding the rest. This will allow Kiva to easily integrate the model in their operation (by simply adding the 3-5 new questions to their loan application form).

1.2.5 Joining The Predicted Poverty To Existing Kiva Data

In this phase, we will find an appropriate way to join the results of the poverty prediction model to the existing Kiva loans data set. We will join the poverty scores by borrower gender and region.

1.3 Metrics

We will utilize the following evaluation metrics for the household income prediction:

- Mean squared error:

$$MSE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \hat{y}_i)^2 \tag{2}$$

- R2 score:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n_{samples}-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n_{samples}-1}(y_i - \bar{y}_i)^2} \tag{3}$$

- Mean squared logarithmic error:

$$MSE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (log_e(1 + y_i) - log_e(1 + \hat{y}_i))^2 \tag{4}$$

- Explained variance score:

$$EV(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}} \tag{5}$$

As evident by our choice, we want to focus on two metrics for the errors in our model and two metrics to help us understand the model's robustness and potential future performance. The mean squared error and the mean squared logarithmic error will allow us to assess the deviation of our model's prediction from the target variable.

We will also utilize the R2 score (the coefficient of determinations) - a mtric that represents the future performance of the model (how accurately new samples are expected to be predicted by the same model). Using this metric will allow us to build a robust model that could generalize well on new data.

The explained variance score will allow us to assess how our model accounts for the deviation from the mean within our sample. This is important for us as we are trying to predict a spectrum of poverty. Or model is expected to account for the nuances of our target variable in the various regions where data is available. Therefore, we have to take the explained variance score into account as we refine our model.

2 Analysis

2.1 Data Exploration

We are using two main data sets: the Kiva loans dataset and the Philippines Family Income and Expenditure Survey (FIES). Both of the datasets are not immediately suitable for the tasks laid out in the Problem Statement. This means that we will have to transform the data in certain ways. We will now look at each data set and its corresponding data transformation.

2.1.1 Kiva Loans Data Set

The Kiva loans data set contains 671205 records for loans granted by Kiva in the last 2 years. For the purpose of this project we will use the following columns:

- Id: The unique identifier for each loan record. It has a numeric value.
- Country: The name of the country where the loan was granted in camel case. We will use this column to select all records from the Philippines.
- Region: Unstructured representation of an administrative entity that is below the level of a country. The entity could be a city, city and a province name separated by comma, village name, other geographic entities or no value. We will use this column to derive the actual administrative region in the Philippines where the loan was granted. The derived administrative region will be further used to join the poverty score by region.
- Borrower_gernders: The unique categories in this column are male and female. However, the columns values could be any combination of those two that is as long as the number of borrowers (e.g. female, female, male for 3 borrowers of the same loan record). We use this column to derive a borrower gender female one hot encoded column. The new column will be subsequently used to join the poverty score by gender.

Here is a sample of the contents of the above-mentioned columns for 5 records:

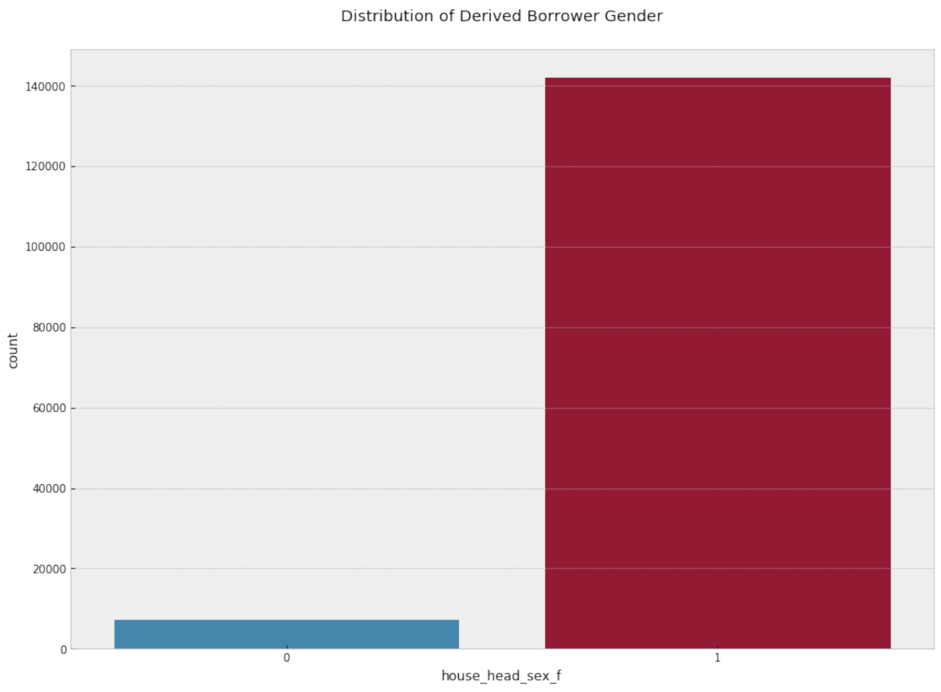
id	country	region	borrower_genders
1209550	Philippines	Dipolog -Zamboanga del Norte	female
888025	Nicaragua	Masaya	female, female, female
1171969	Philippines	Roxas Palawan	female
867403	El Salvador	NaN	male
742400	Nigeria	Kaduna	female

As we can see from this short example, the region column could contain a ¡city¿ ¡province¿ tuple (for loan id 1209550, where the city is Dipolog and the province is Zamboanga del Norte), only city (Masays for loan id 888025), municipality name (Roxas Palawan for loan id 1171969) or no valid value (NaN for loan id 867403). We will need to transform this column to a standard value using matching tables and other heuristics.

For the records from the Philippines, the unique values for the borrower_genders column are only: female, male and NaN. Thats why will also transform the borrower_genders column by mapping the female category to 1 and the male category to 0. There are only 80 records with missing values (NaN), so we could do away with them.

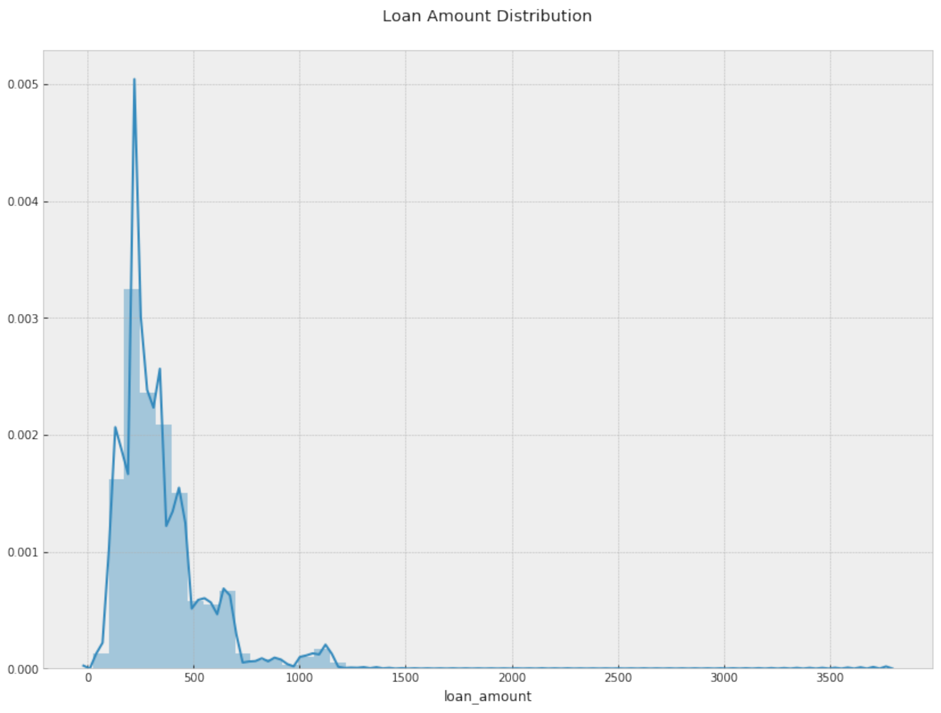
Let us now take a look into some key characteristic distributions. The borrower gender distribution seems to be skewed strongly in favor of female borrowers:

Figure 1: Number of Loans Per Region



The loan amount distribution is positively skewed with most of the weight at around 200 - 250 PHP.

Figure 2: Number of Loans Per Region



2.1.2 Philippines Family Income and Expenditure Survey

The Philippines Family Income and Expenditure Survey (FIES) contains 41544 records of household incomes, expenditure, living conditions and other affluence indicators. The data set contains 59 columns representing information in the following categories:

- Demographics
  - Agricultural Household indicator
  - Household Head Sex (categorical)
  - Household Head Age (categorical)
  - Household Head Marital Status (categorical)
  - Type of Household (categorical)
  - Total Number of Family members
  - Members with age less than 5 year old
  - Members with age 5 - 17 years old
  - Region (categorical)
  - Household Head Highest Grade Completed (categorical)
- Expenses
  - Total Food Expenditure
  - Bread and Cereals Expenditure
  - Total Rice Expenditure
  - Meat Expenditure
  - Total Fish and marine products Expenditure
  - Fruit Expenditure
  - Vegetables Expenditure
  - Restaurant and hotels Expenditure
  - Alcoholic Beverages Expenditure
  - Tobacco Expenditure
  - Clothing, Footwear and Other Wear Expenditure
  - Housing and water Expenditure
  - Medical Care Expenditure
  - Transportation Expenditure
  - Communication Expenditure
  - Education Expenditure
  - Miscellaneous Goods and Services Expenditure
  - Special Occasions Expenditure
  - Crop Farming and Gardening expenses
- Income

  - Main Source of Income (categorical)
  - Total Income from Entrepreneurial Activities
  - Household Head Job or Business Indicator (categorical)
  - Household Head Occupation (categorical)
  - Household Head Class of Worker (categorical)
  - Total number of family members employed

- Living Conditions
  - Imputed House Rental Value
  - Type of Building/House (categorical)
  - Type of Roof (categorical)
  - Type of Walls (categorical)
  - House Floor Area
  - House Age
  - Number of bedrooms
  - Tenure Status (categorical)
  - Toilet Facilities (categorical)
  - Electricity
  - Main Source of Water Supply (categorical)
  - Number of Television
  - Number of CD/VCD/DVD
  - Number of Component/Stereo set
  - Number of Refrigerator/Freezer
  - Number of Washing Machine
  - Number of Airconditioner
  - Number of Car, Jeep, Van
  - Number of Landline/wireless telephones
  - Number of Cellular phone
  - Number of Personal Computer
  - Number of Stove with Oven/Gas Range
  - Number of Motorized Banca
  - Number of Motorcycle/Tricycle

It is worth noting that the fields Household Head Occupation and Household Head Class of Worker contain only 34008 non-NULL values. We should also apply some category reduction to the fields, one hot encoding as well as query-friendly names to these fields. In addition, all the categorical fields (marked as 'categorical' in the bullet list above) will require additional transformations. Besides the categoricall transformations, all column names need to be changed so that they are query-friendly.

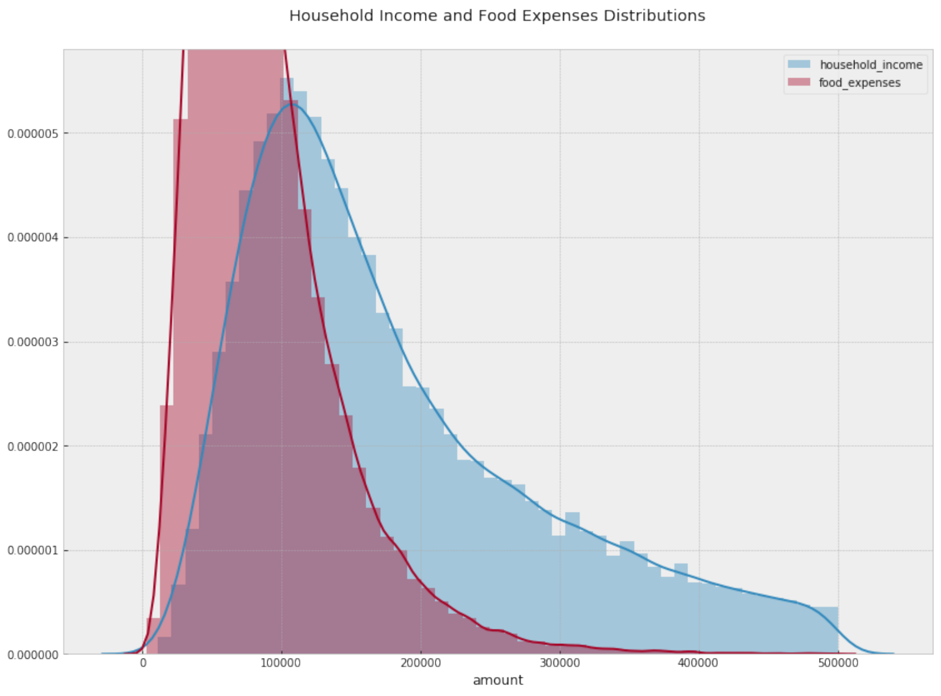
It is obvious that this dataset will require a significant number of preprocessing modifications. We can group those in the following preprocessing operations:

- Mapping column string values to query-friendly names
- One hot encoding
- Columns merging
- Reducing / categorizing unique column values
- Renaming columns to query-friendly names

We will look at all those preprocessing operations in detail in the Data Preprocessing chapter.

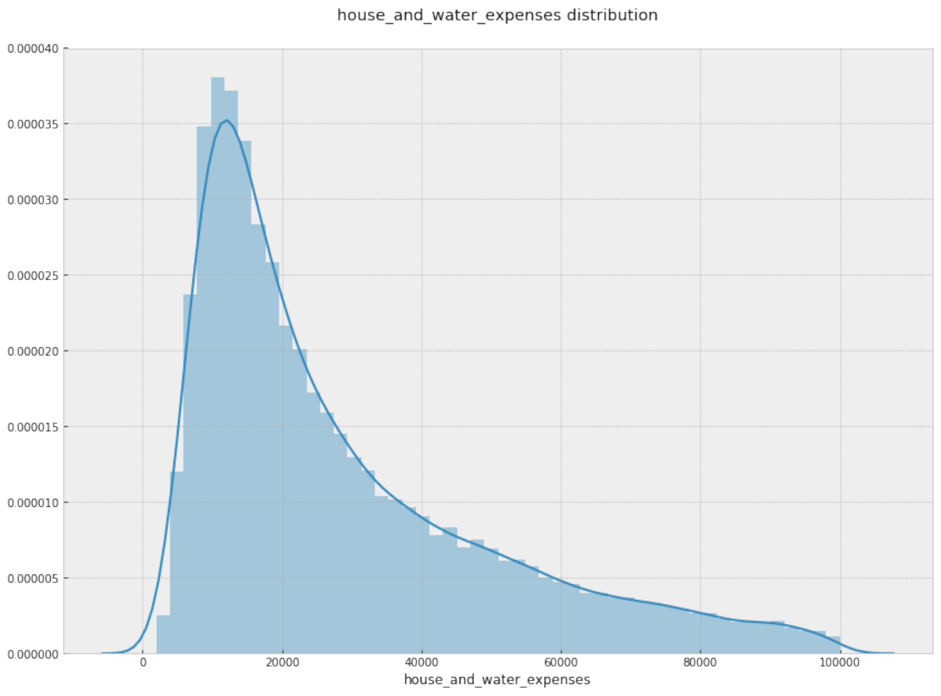
Let us now take a look into some key characteristic distributions. The distributions of the food expenses and household income are both positively skewed. However, the distribution of the household income is offset with regards to the food expenses (as expected, the income is mostly higher than the expenses):

Figure 3: Number of Loans Per Region



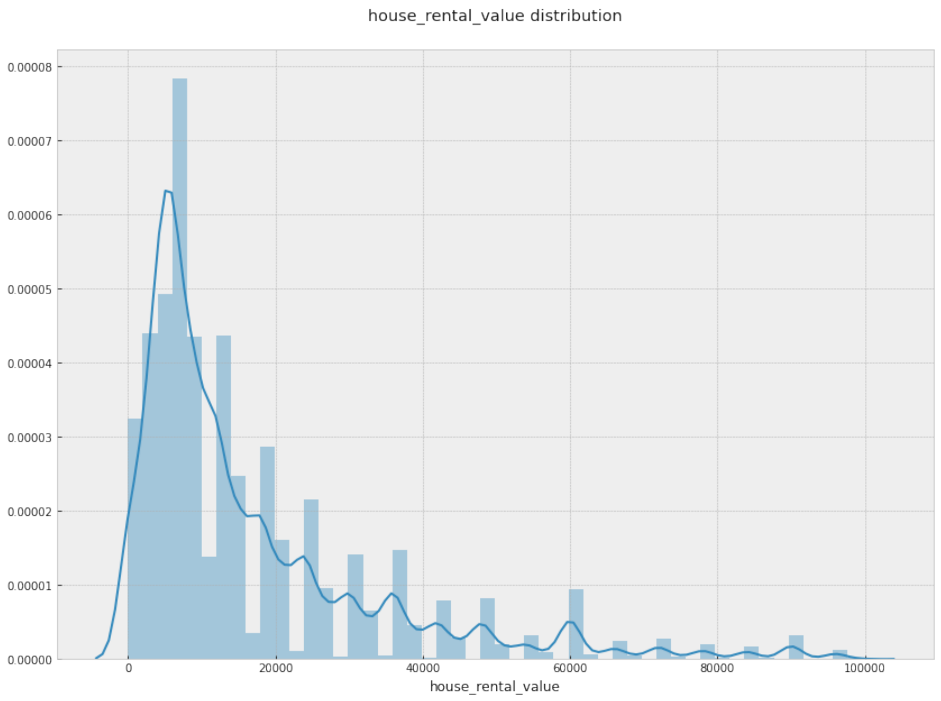
Next, we can look at another important category - house and water expenses. The distribution is again positively skewed with most of the weight at around 150000 PHP:

Figure 4: Number of Loans Per Region



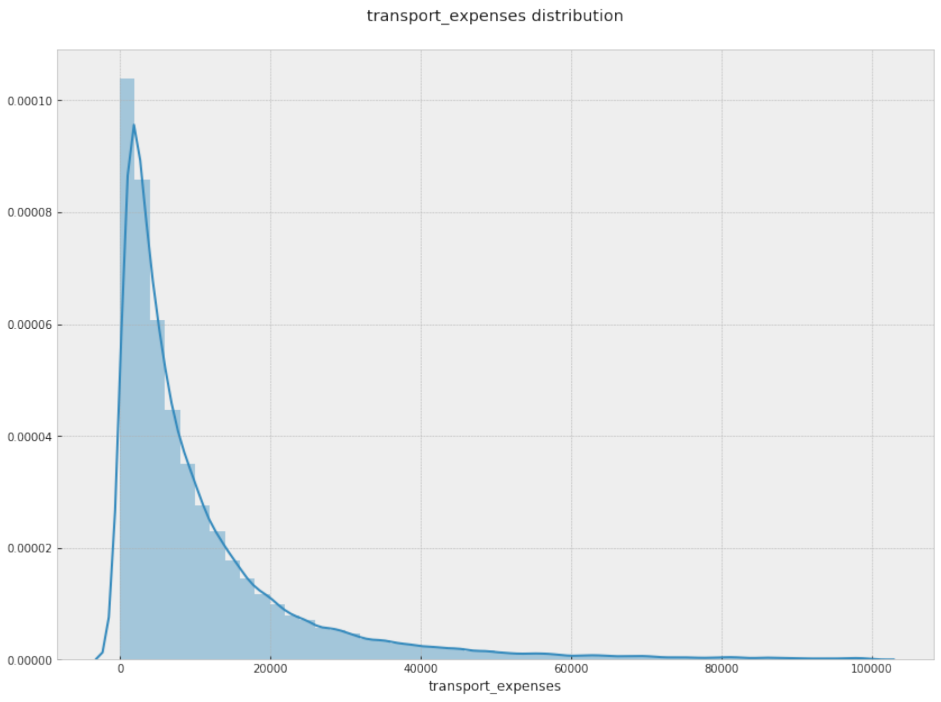
Another interesting feature is the house rental value. Here, since rental prices are more or less round number, some discrete steps of the distribution density can be observed:

Figure 5: Number of Loans Per Region



The transport expenses variable is positively skewed with most of its weight between 0 and 2000 PHP. This should alert us to the fact that some of the poverty in the Philippines could be caused by its island terrain and consequently the isolated nature of some regions.

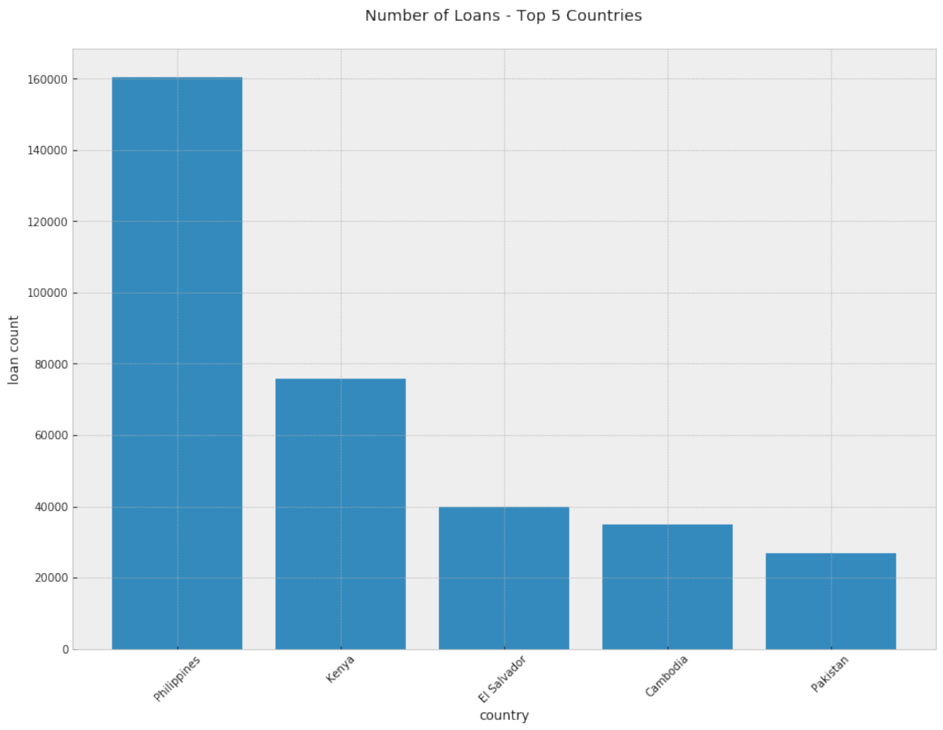
Figure 6: Number of Loans Per Region



2.2 Exploratory Visualization

Let us start with the most important question - is the Philippines the region with the highest count of Kiva loans. To answer it, we will group and count the loans by country, sort the results by country and display the 5 countries with the highest loan counts:

Figure 7: Number of Loans Per Region

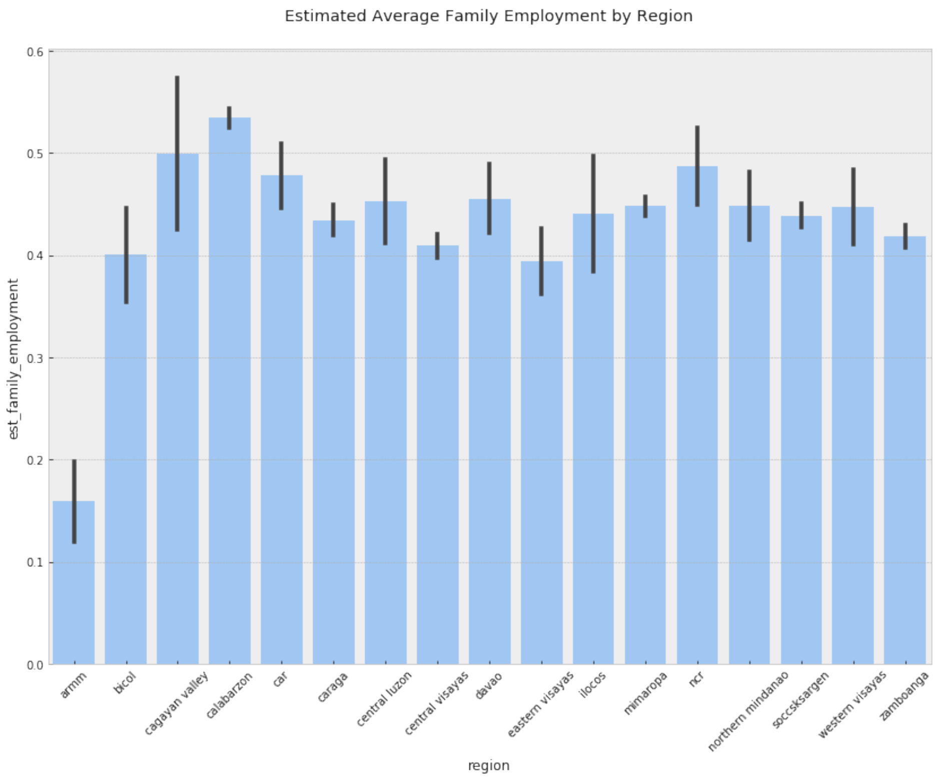


As we can see, the Philippines is the region where Kiva mostly focuses their efforts in terms of loan count. Actually, when it comes to loan count, there are twice as many loans granted in the Philippines as in the second country in the list (Kenya).

Now we can focus on the Philippines and the FIES data. We will first derive an estimated family employment feature. It will be defined as: number of employed family members / (total number family members - children). We will also derive no_electricity and no_running_water indicators.

Let's now plot the estimated family employment by region:

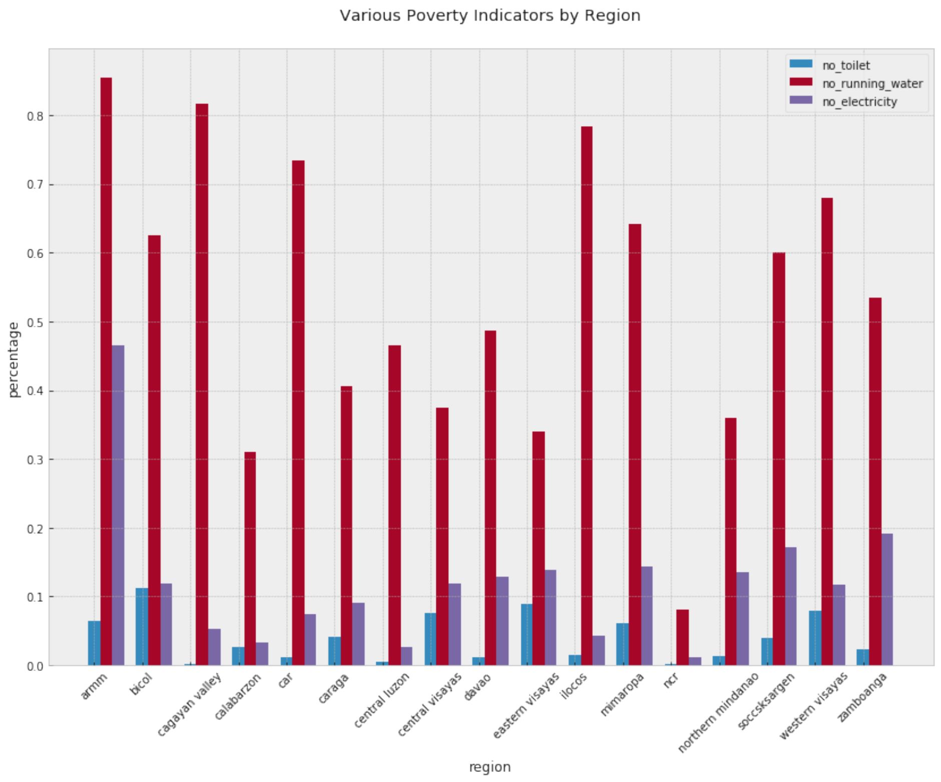
Figure 8: Estimated Employment Per Region



We can see from the very beginning that the Autonomous Region of Muslim Mindanao (ARMM) is severely affected by unemployment. Another notable feature is that in the Cagayan Valley the disparity in employment between households with male and female heads is greatest.

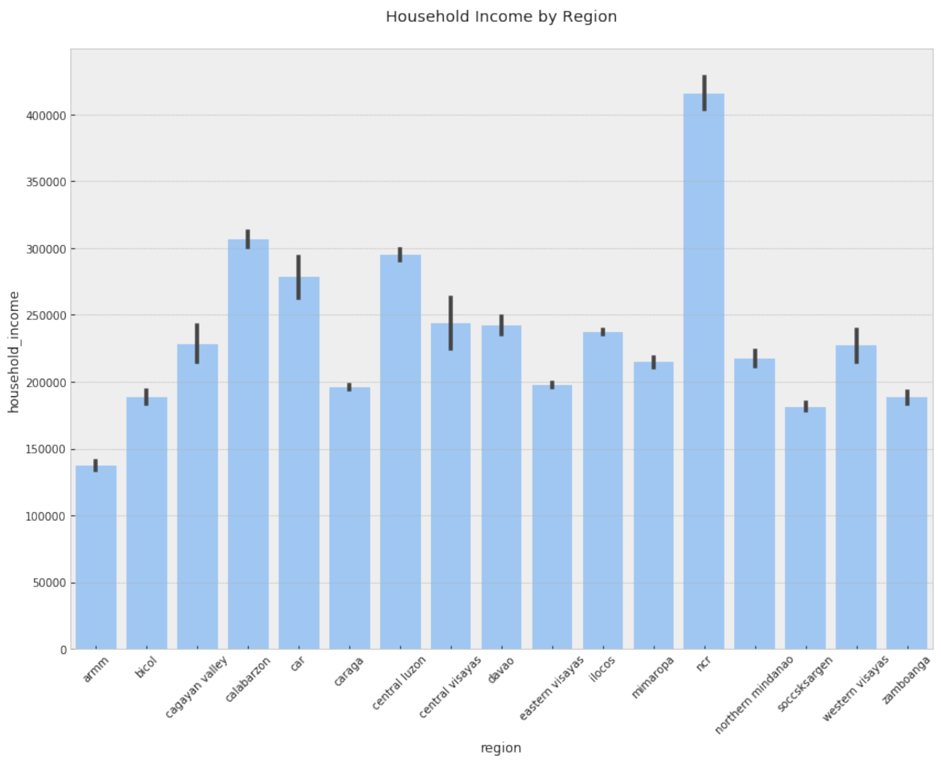
We continue by visualizing the three major poverty indicators (no running water, no toilet and no electricity):

Figure 9: Poverty Indicators Per Region



Again ARMM is severely affected as measured by all three indicators. A good sense check of our data is the inspection of the corresponding values in the National Capital Regions (NCR). We should expect that there, the lack of infrastructure (running water, electricity and toilet) is the lowest. That is confirmed by the chart. We could also expect that incomes in the ARMM province would be the lowest for the country. Let's confirm that:

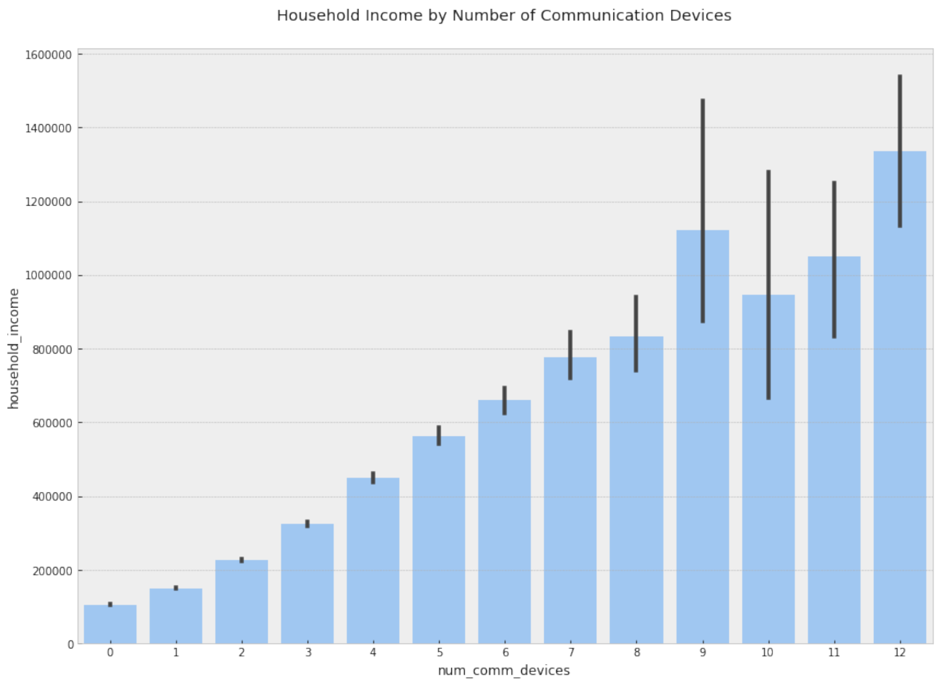
Figure 10: Household Income Per Region



The income distribution by region fits our expectations. Another observation (for the sake of sense-checking the data) is that the National Capital Region (NCR) has the highest average income. This is also in line with expectations.

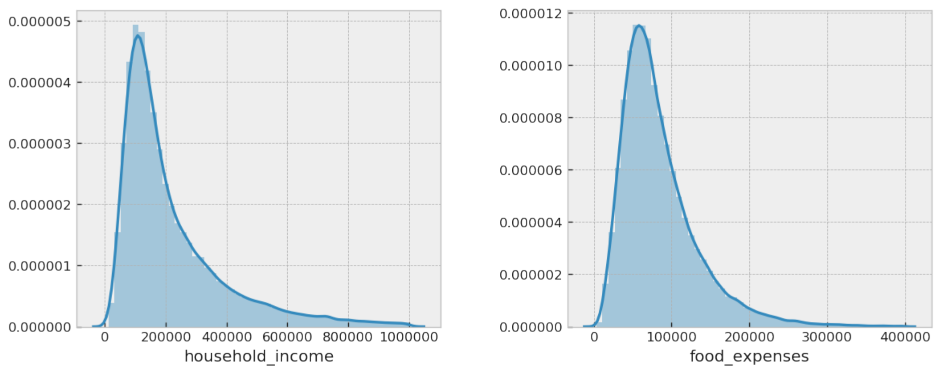
We could expect that lack of communication means lack of access to the economy of the country. Let's investigate the relationship between the number of communication devices and household income in the next plot:

Figure 11: Number of Communication Devices with Household Income



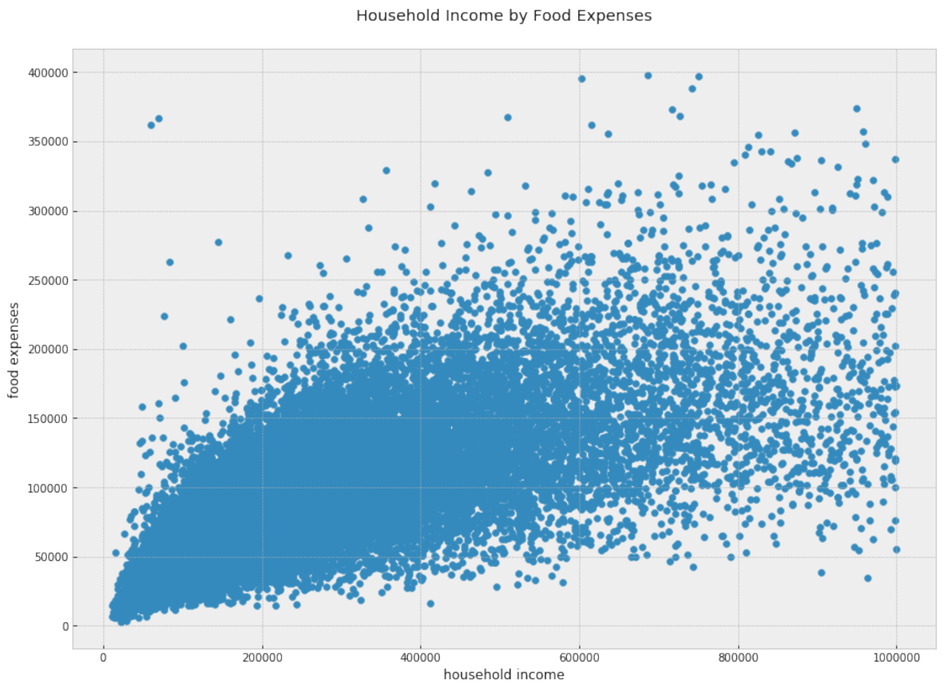
Let's now turn to more general data patterns and investigate for useful predictors of income. We will start by comparing the distributions of food expenses and household income:

Figure 12: Distribution of Food Expenses and Household Income



Apparently, the distributions are quite similar (both are positively skewed). We can visually inspect the correlation between the two variables in the next scatterplot:

Figure 13: Household Income by Food Expenses



There is an apparent correlation between household income and food expenses.

2.3 Algorithms and Techniques

We utilize the following data selection and transformation techniques in the poverty scores modeling work:

- Correlation inspection We use Pandas implementation of the Pearson correlation coefficient to inspect and reduce the number of input features to the model. The correlation coefficient is defined as:

$$r = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2} \qquad (6)$$

- Feature weight inspection: We use a models feature weight method in order to obtain the significance of the important features. This will allow us to reduce the number of new data that Kiva will need to obtain if they are to implement our model. Ideally, we have to reduce the number of features to 3-5 important questions that can be asked in the field.
- Normalization: As we saw in the visualization chapter, the household income distribution is positively skewed. On the other hand, there are many one hot encoded columns as explained in the Data Preprocessing chapter. This means that we will have data (such as the income and expenses) that will be overpowering the 0-1 one hot encoded indicators due to its scale. Thats why we need to apply a normalization on the dataset.

In our modeling work, we do not want to ignore any possibility for producing good results. Thats why we will use as many different model classes as possible. This will give us the opportunity to select the best performing model and tweak its parameters as well as the chance to use certain models idiosyncrasies (e.g. the decision tree model can be visualized, the linear regression model can be converted to a scorecard, etc.). We utilize the following modelling algorithms:

- AdaBoost Regressor: Boosting algorithms generally work well out of the box and we will use this one as representative from its class.
- Support Vector Machines Regressor: Support vector machines allow us to find data trends in higher dimensions and translate them back to our existing feature space.

- Linear Regression: Linear regressions are simple and hold great explanatory power.
- Decision Tree Regressor: Decision trees can be visualized well and are not affected by outliers.
- Multi Layer Perceptron Regressor: This algorithm and its SciKit Learn implementation provide us with a quick and easy neural network class algorithm.

Let us now discuss how each of the models work. We will start with the AdaBoost regressor. The AdaBoost algorithm relies on using small decision tree stubs (or boolean expressions) to make simple cuts in the distribution of the input variables. It then assigns these segments values. Upon evaluation (at the end of each epoch), if mistakes are encountered their weight is amplified and a new process of creating stubs starts where the focus is on those amplified errors. At the end of that epoch, the algorithm evaluates itself and finds new errors. Those are in turn aplified. The process continues until convergence.

The linear regression algorithm attempts to fit a line between a series of points. This is achieved by finding a line for which the total mean squared error for all points in the development sample is the smallest. The line's equation can then produce values for other points missing in the sample (predict). This algorithm's main weakness is its linearity - it cannot be applied on data trends of higher order.

The support vector machines algorithm attempts to maximize the distance between zones with different values of the target variable. This is achieved by dividing them and maximizing the distnace between the vectors formed by the outer-most points of each category (a.k.a 'support vectors'). This is an improvement upon the method of the linear regression where a line could be drawn in many different ways through several points. With the SVMs the maximization of the distance between the support vectors guarantees that the line between the various categories (values) is drawn in an optimal way.

The decision tree regressor attempts to split the input features into zones of values of the target variable (much like the AdaBoost regressor). The difference is that splits are done several levels done. Each split represents a binary question (e.g. 'are the food expenses below 10000 PHP?'). Each zone could thus be broken down into smaller and smaller areas until the algorithm fits well the distribution of points with different values.

The Multilayer perceptron is SciKit Leanr's implementation of a neural network. Each perceptron is a digitally simulated neuron - it has several inputs (that mimic a neuron's dendrites) and an output (that mimics the neuron's axon). Upon certain signal received by the dendrites, the neuron activates and sends a signal down the axon. In the case of the perceptron, when the weighted sum of the inputs crosses a certain threshold value, the perceptron outputs 1 or 0. Depending on its activation function, the perceptron could output a probability of 1. In the case of the multilayer perceptron, the output of several perceptrons could become the input for another perceptron. Thus a network is formed between several groups of perceptrons that receive their input from another group of perceptrons. Each group is thus called a 'layer'. By building a neural network with many layers (a.k.a. deep neural network), the algorithm can abstract concepts on ever higher levels. For example, a convolutional neural network at its first layer could perceive edges, one level deeper it can perceive primitive shapes (straight lines, diagonal lines, circles). Still one layer further it could perceive objects such as eyes, noses and ears. One layer furhter, che neural network could perceive faces. For our specific case, the multilayer perceptron is not a convolutional neural network but could abstract numirc data to higher levels.

2.4 Benchmark

Since the normalized values for the household income have a range between 0 and 1, we will use 3 dummy predictors to evaluate the performance of our actual model against:

- A dummy predictor that always predicts household income 0
- A dummy predictor that always predicts household income 1
- A dummy predictor that always predicts household income 0.5

We will evaluate these regressors based on our selected evaluation metrics and compare them to our actual model under development.

3 Methodology

3.1 Data Preprocessing

As we saw in the data exploration chapter, there are several preprocessing steps we have to take in order to make the data usable in our modeling task:

- Map the Kiva loans region column to actual regions in the Philippines
- Transform the categorical fields from the FIES data to one hot encoded fields
- Give all columns query-friendly names
- Where necessary, reduce the categories of certain columns from the FIES data set
- Where necessary, merge certain columns from the FIES data set
- Normalize all columns from the FIES data sets

Here, well describe in detail the pre-processing of both the Kiva loans and the FIES data sets.

3.1.1 Kiva Loans Data Set Transformation

Let us first lay out the administrative region hierarchy of the Philippines:

1. Island Group
2. Region
3. Province
4. City

For most of the cases (about 145000 out of 160000), the region column from the Kiva loans data set contains a pattern of the type: city name, province. We can make a supplementary matching table with the administrative region types as columns and the administrative region names as rows. We will then separate each city name, province entry into city name and province name. We will then use the province name to match the actual Region in the Philippines.

Here is an example of a few rows from our matching table:

island_group	region	province	city
luzon	ilocos	ilocos norte	laoag
luzon	ilocos	ilocos sur	vigan
luzon	ilocos	la union	san fernando
luzon	ilocos	pangasinan	lingayen
luzon	cagayan valley	batanes	basco
luzon	cagayan valley	cagayan	tuguegarao city
luzon	cagayan valley	isabela	ilagan

If we have the following entry from the Kiva loans data set: laoag, ilocos norte, we will separate it into laoag and ilocos norte, take the province name of ilocos norte and look up what is its adjacent region (in this case ilocos). That means that our matching algorithm for the bulk of the entries is:

1. Convert all names to lowercase
2. For each entry in the region column from the Kiva loans data set:

- Split the entry into city and province
- Take the province
- Match the province with a region from the matching table using Pythons built-in get_close_matches method
- Return the region from the matching table

For the cases where this method is unsuccessful (15000 cases), we will attempt to match the city name. We will do that by using the algorithm above, but matching against the city instead of the province. This helps us match further 5000 entries from the Kiva loans table.

We will discard all entries that have missing value, no major population center listed or have unclear location (about 10000 rows).

3.1.2 FIES Data Set Transformation

There is one general transformation that all columns need to undergo: renaming them to query-friendly names. In addition, before one hot encoding, all categorical string values will also be mapped to query-friendly names as those values will become one hot encoded columns themselves. It is worth describing our query-friendly naming convention:

- All words are in lowercase
- There are no spaces, words are separated by an underscore (_)
- Where possible, words are abbreviated

Here is one example of converting a column name: 'Members with age less than 5 year old' becomes 'num_children_younger_5'. Here is an example of categorical string values mapping (for the 'Type of Household' column): the value 'Extended Family' is mapped to 'house_ext_family'. When one hot encoded, the value 'house_ext_family' will become a column itself containing the values 1 and 0.

Let us now go over all transformed columns and describe their transformations in detail:

- Region: the values are mapped to a common naming convention shared by the Kiva loans data set.
- The columns: 'Bread and Cereals Expenditure', 'Total Rice Expenditure', 'Meat Expenditure', 'Total Fish and marine products Expenditure', 'Fruit Expenditure', 'Vegetables Expenditure' are removed (only the food_expenses column is used instead).
- The values of the columns: 'Restaurant and hotels Expenditure', 'Alcoholic Beverages Expenditure', 'Tobacco Expenditure' are summed into the 'non_essential_expenses' column. The columns: 'Restaurant and hotels Expenditure', 'Alcoholic Beverages Expenditure', 'Tobacco Expenditure' are then removed from the data set.
- The categorical string values from the 'Household Head Sex' column are mapped to: 'Female' : 1 and 'Male' : 0. The column is then renamed to: 'house_head_sex_f'.
- The column 'Household Head Marital Status' contains various categorical string values such as: 'Single', 'Widowed', 'Divorced/Separated', 'Annulled', 'Unknown', Married, etc. We will reduce these statuses to 2 values: 'house_head_single' and 'house_head_partner'. This is because for the purpose of household income estimation, what matters is if there are two income earners in the household or just one. We will then one hot encode the column and drop it.
- The 'Household Head Highest Grade Completed' contains many statuses corresponding to each possible grade of school completed as well as to all possible university majors. This is quite an extensive list, but for the purpose of income estimation, only the education level matters (e.g. a person with university education will definitely earn more than an illiterate person). Thats why we will group all values into: 'house_head_illiterate', 'house_head_primary_ed', 'house_head_secondary_ed', 'house_head_tertiary_ed'. We will then one hot encode these grouped values and drop the column.
- The column 'Household Head Job or Business Indicator' can be renamed to 'house_head_empl' and the values can be mapped as: 'With Job/Business' : 1, 'No Job/Business' : 0.

- The 'Household Head Occupation' column contains over 300 values. This is too specific and will unnecessary complicate our data set if one hot encoded. Besides many job types are more or less duplicates (e.g. General Manager in the Hospitality Industry and General Manager in the Textile Industry). On the other hand, our ultimate goal is to be able to estimate poverty levels. With regards to that, we can derive a flag for all professions that are related to farming. Our intuition is that in the context of poverty assessment, the farming professions would be meaningful. The rationale behind that is that farmers are usually the most isolated and poor individuals, especially in developing countries. We'll use a simple regular expression to derive the farming professions and map them to a 'house_head_farmer_flag'.
- For the columns: Type of Household, Type of Building/House, Main Source of Income, 'Type of Roof', 'Type of Walls' and 'Toilet Facilities': the categorical values are mapped to query-friendly names and one hot encoded.
- The column 'Main Source of Water Supply' is reduced to an indicator of 'running_water'. The indicator has values 1 and 0.
- The values of the columns: 'Number of Television', 'Number of CD/VCD/DVD', 'Number of Component/Stereo set', 'Number of Personal Computer' are summed into the column 'num_electronics'. The columns: 'Number of Television', 'Number of CD/VCD/DVD', 'Number of Component/Stereo set', 'Number of Personal Computer' are then removed from the data set.
- The values of the columns: 'Number of Landline/wireless telephones' and 'Number of Cellular phone' are summed into the column 'num_comm_devices'. We are summing those two columns into a separate feature from the 'num_electronics' because poverty is also related to lack of access to communications and to the financial system. It is apparent (as shown in the Visualization chapter) that lack of communication is a meaningful feature and should be derived on its own. The columns: 'Number of Landline/wireless telephones' and 'Number of Cellular phone' are then removed from the data set.
- The columns: 'Number of Car, Jeep, Van', 'Number of Motorized Banca', 'Number of Motorcycle/Tricycle' are summed into the column 'num_vehicles'. The columns 'Number of Car, Jeep, Van', 'Number of Motorized Banca', 'Number of Motorcycle/Tricycle' are then dropped from the data set.
- All remaining columns are renamed to query-friendly names.

As an additional step to our modeling, we will also normalize the so transformed FIES data set. So that skewing or uneven scaling of the data wont affect the performance of our models.

3.1.3 Data Selection

As a final step in our preprocessing we need to inspect the various feature correlations. We will start with columns from the FIES data set which have similar meaning to the original target variable. There are several columns that would correlate highly with the 'household_income' column. Those are:

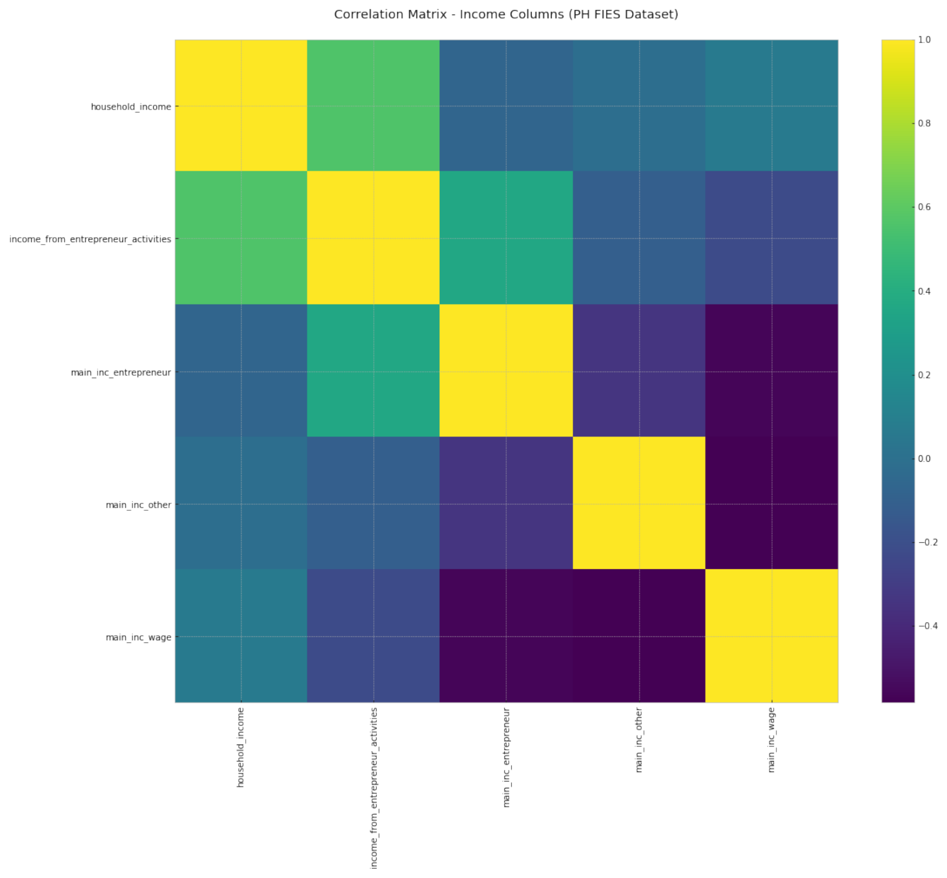
- 'income_from_entrepreneur_activities'
- 'main_inc_entrepreneur'
- 'main_inc_other'
- 'main_inc_wage'

Let's see the actual correlation levels and confirm that:

feature	household_income	income_from_entrepreneur_activities 1	main_inc_entrepreneur	main_inc_other	main_inc_wage
household_income	1	0.563662	-0.068279	-0.010181	0.067958
income_from_entrepreneur_activities	0.563662	1	0.363651	-0.106587	-0.220697
main_inc_entrepreneur	-0.068279	0.363651	1	-0.341511	-0.564373
main_inc_other	-0.010181	-0.106587	-0.341511	1	-0.583149
main_inc_wage	0.067958	-0.220697	-0.564373	-0.583149	1

Here is a visualization of the table above:

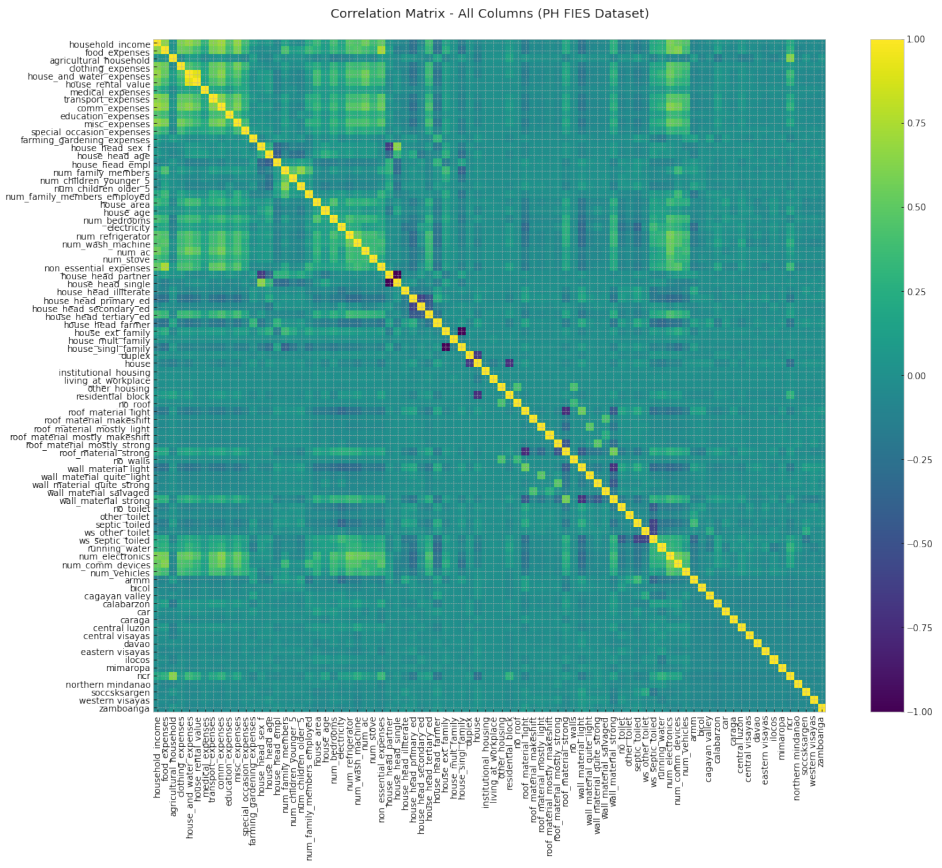
Figure 14: Correlation Matrix - Income Features



As we can see, there is a high correlation between the various income columns and the household_income target. Thats why we have to remove these columns from the data set.

We can now plot all remaining variables and visually inspect for any other correlation patterns:

Figure 15: Correlation Matrix - All Features



As we can see, there are some high positive and some inverse correlations. Let's zoom in on several examples:

- The expenses columns seem to correlate highly with one another. That is reasonable as high 'transportation' or 'other' expenses indicate an affluent household. We could expect such a household to also have high 'clothing' expenses.
- There are some inversely-correlated columns (e.g. house_head_single and house_head_parther). The relationship there is more than obvious - if the house head is single, they definitely don't have a partner.
- The household_income column seems to correlate highly with the following fields:
  - Expenses (food expenses, clothing expenses, house and water expenses, transport expenses, communication expenses and miscellaneous expenses)
  - House rental value
  - Number of ACs (air conditioning appliances)
  - Number of electronic devices (DVDs, CD, stereo, etc.)
  - Number of communication devices (landline or mobile phones)
  - Number of vehicles (cars, tricycles, motorized boats)

Those correlations seem to reasonably point towards the welfare of individuals. After all, poverty is not only the lack of disposable income, but also exclusion through lack of connectivity, transportation and financing.

At this point, we will:

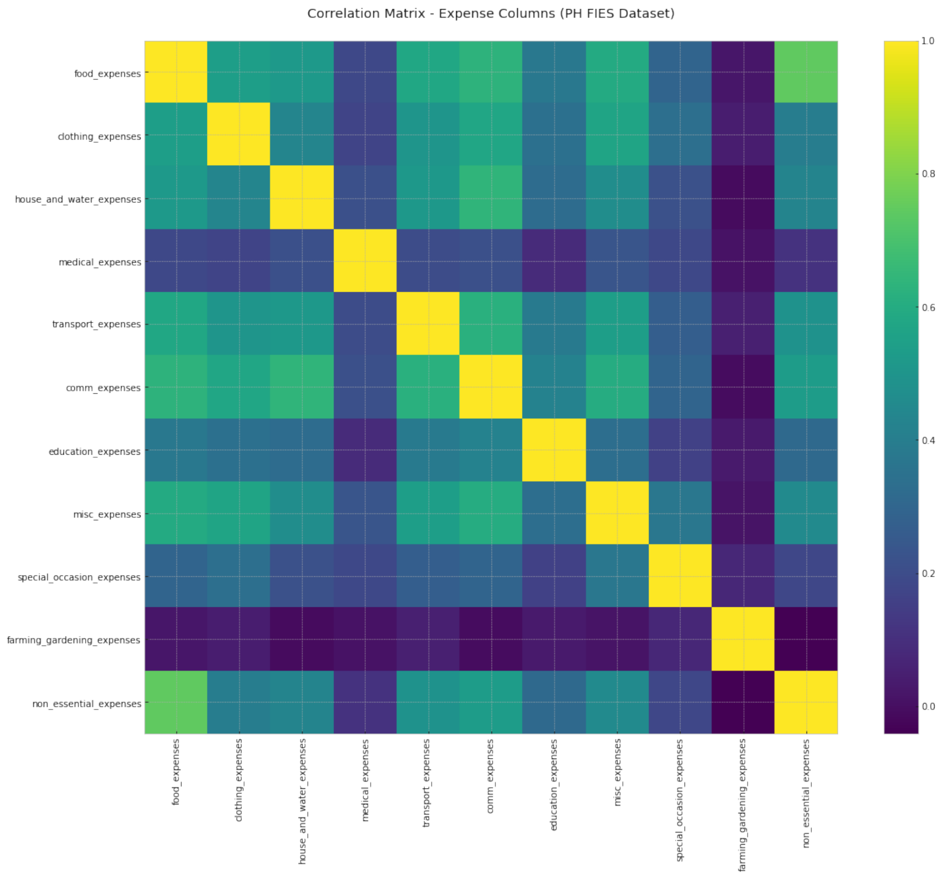
- Visualize the correlation of the expenses features:
  - food_expenses

  - clothing_expenses
  - house_and_water_expenses
  - medical_expenses
  - transport_expenses
  - comm_expenses
  - education_expenses
  - misc_expenses
  - special_occasion_expenses
  - farming_gardening_expenses
  - non_essential_expenses
- Remove one of each inversely-correlating pairs
- Remove the non-relevant expenses. The reason for that is to make the model less reliant on information from the same category. This will also allow us to require less data if we scale the model internationally (it will be way easier to find food expenses data alone than food, clothing, miscellaneous, transportation, etc. together).

feature	food_expenses	clothing_expenses	house_and_water_expenses	medical_expenses	transport_expenses	comm_expenses	education_expenses	misc_expenses	special_occasion_expenses	farming_gardening_expenses	non_essential_expenses
food_expenses	1	0.543237	0.522252	0.186645	0.577372	0.633819	0.374898	0.594453	0.296624	0.0208	0.744933
clothing_expenses	0.543237	1	0.432092	0.172845	0.503026	0.574041	0.34377	0.565225	0.337542	0.04438	0.401855
house_and_water_expenses	0.522252	0.432092	1	0.212181	0.515797	0.639698	0.327944	0.470872	0.216761	-0.01216	0.429699
medical_expenses	0.186645	0.172845	0.212181	1	0.20037	0.214714	0.088768	0.231703	0.185885	0.010141	0.113158
transport_expenses	0.577372	0.503026	0.515797	0.20037	1	0.621611	0.386692	0.54453	0.272043	0.050873	0.491167
comm_expenses	0.633819	0.574041	0.639698	0.214714	0.621611	1	0.419618	0.604349	0.293515	-0.005839	0.533568
education_expenses	0.374898	0.34377	0.327944	0.088768	0.386692	0.419618	1	0.334283	0.15884	0.035074	0.315932
misc_expenses	0.594453	0.565225	0.470872	0.231703	0.54453	0.604349	0.334283	1	0.371388	0.013091	0.458687
special_occasion_expenses	0.296624	0.337542	0.216761	0.185885	0.272043	0.293515	0.15884	0.371388	1	0.074255	0.180372
farming_gardening_expenses	0.0208	0.04438	-0.01216	0.010141	0.050873	-0.005839	0.035074	0.013091	0.074255	1	-0.04078
non_essential_expenses	0.744933	0.401855	0.429699	0.113158	0.491167	0.533568	0.315932	0.458687	0.180372	-0.04078	1

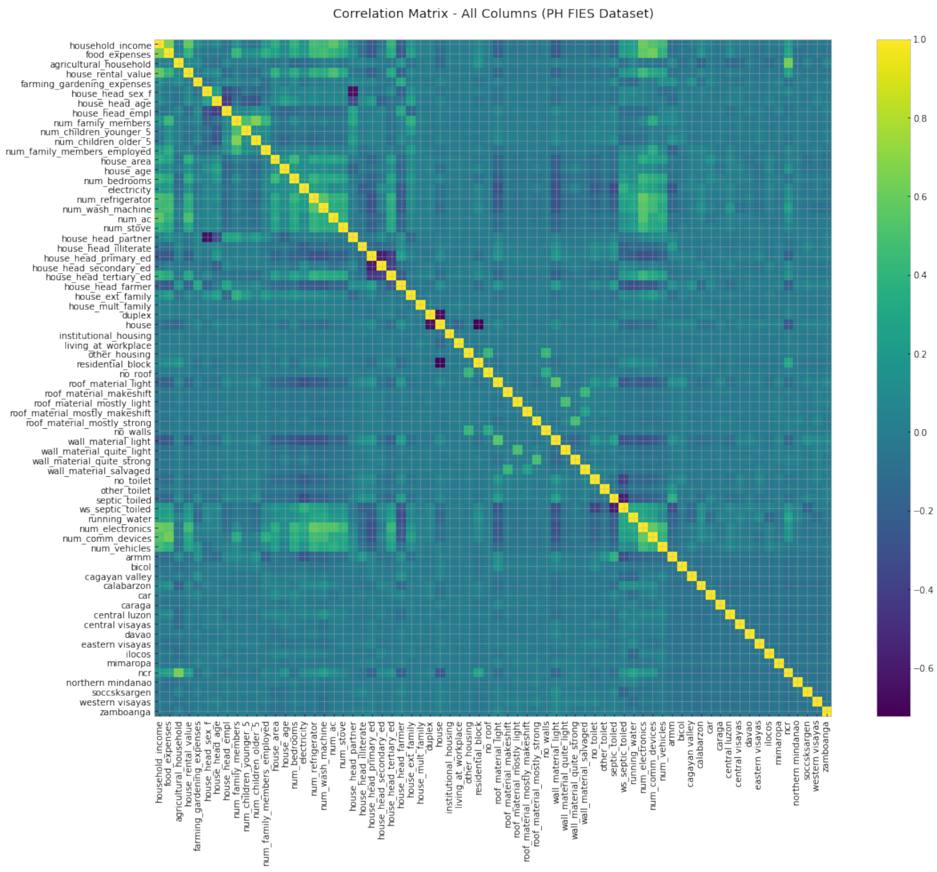
Here is a visualization of the correlation matrix:

Figure 16: Correlation Matrix - Expenses Features



We will remove all expense columns except the food_expenses. Here is a visualization of the correlation between the remaining features:

Figure 17: Correlation Matrix - All Features



3.2 Implementation

At this point, we can derive an initial model. We'll use that model and its highest-performing features in order to reduce the number of input columns later on.

We will separate the data into a training and testing sets. We will only use the training set with cross validation in order to select the best fields. We will then use the test set to evaluate the final model's performance. The specific steps we will p

1. Scaling and normalizing the data
2. Dividing the data into a training and testing sets
3. Evaluating an initial model (out of all the candidates described in the Algorithms section)
4. Selecting the columns with the highest weights
5. Creating the final model
6. Measuring the model's performance

We will train and validate several model types without any tweaking. We'll do that on portions of the training dataset. We'll then produce some performance metrics that will allow us to focus on one or two of the models. For the purpose of benchmarking, we will also include validation results for three dummy regressors that always predict 0, 0.5 and 1 respectively:

regressor	mean_squared_error	r2_score	mean_squared_log_error	explained_variance_score
ada_boost	0.000585	0.881831	1.62E-04	8.89E-01
svr	0.006065	-0.224031	1.75E-03	8.13E-01
sgd	0.001681	0.660628	5.07E-04	6.61E-01
decision_tree	0.000024	0.99521	8.00E-06	9.95E-01
linear	0.000425	0.914236	1.32E-04	9.14E-01
mlp	0.000156	0.968477	5.30E-05	9.68E-01
dummy_0.0	0.797975	-160.057695	4.06E-01	0.00E+00
dummy_0.5	0.157458	-30.780255	0.054666	0.00E+00
dummy_1	0.016941	-2.419269	0.004708	-2.22E-16

As we can see, the Decision Tree and the Multi Level Perceptron regressor, the Linear regression and AdaBoost regressor are performing best out of the box. All of them outperform the dummy regressors in all metrics by a significant margin.

Let's look the top 4 models feature importances (or in the case of the Linear Regression and the Multi Layer Perceptron regressor, the feature coefficients):

AdaBoost regressor

feature	importance
food_expenses	0.744377
farming_gardening_expenses	0.198619
house_rental_value	0.047395
agricultural_household	0.006434
house_area	0.001426
house_head_partner	0.00063
house	0.000366
armm	0.000361
house_head_empl	0.000353
num_electronics	0.000039

Linear Regression

feature	importance
no_roof	2257.82356
ncr	750.759429
car	694.030488
zamboanga	669.26106
duplex	536.816816
other_toilet	509.406187
num_refrigerator	471.154704
davao	469.813611
house	466.653182
residential_block	434.85659

Decision Tree regressor

feature	importance
food_expenses	0.632135
farming_gardening_expenses	0.348987
house_rental_value	0.016567
house	0.001125
house_head_partner	0.000141
num_comm_devices	0.000097
house_head_primary_ed	0.000077
agricultural_household	0.000073
num_bedrooms	0.000063
num_electronics	0.00006

MLP regressor

feature	importance
farming_gardening_expenses	0.133671
house_area	0.082551
food_expenses	0.0688
num_children_older_5	0.054415
num_comm_devices	0.046787
num_electronics	0.027852
house_head_empl	0.02675
house_head_partner	0.006638
running_water	0.004642
house_head_primary_ed	0.003707

We can draw some conclusions the feature importance rankings, namely:

- The food_expenses variable has the highest weight in both the AdaBoost and the Decision Tree models. This seems to be reasonable as low income families have predominantly food expenses that they need to limit due to their situation.
- The farming_gardening_expenses has the second highest weight in both the AdaBoost and the Decision Tree models. This is also is in line with the general reasoning for poor households - they are mostly rural and within farming communities. Actually the Kiva dataset for loan themes contains a rural percentage column (supposedly for the same reason).
- There are other important indicators of poverty which also seem in line with our intuition. Such are:

  - Number of communication devices (or lack thereof)
  - Agricultural household indicator
  - House area
  - House head employment status
  - House head having partner indicator
  - Number of family members

It's also interesting to note that the Linear Regression has placed high importance on the regions and the living conditions (the no_roof, other_toilet, num_refrigerator). Unfortunately we cannot simply use the region features because that would make the model harder to scale in the future (such a model will be basically a map of poverty-stricken regions).

The MLP regressor (at least in one of the dimensions if its matrix) has selected a mix of features, namely:

- Expenses: farming_gardening_expenses
- Demographics: num_children_older_5, house_head_secondary_ed, house_head_tertiary_ed
- Living conditions: house_ext_family, wall_material_quite_light, residential_block, roof_material_light
- Region: armm (as we saw in the Visualization chapter, this is the poorest region in the Philippines)

Here, we have some features that we can work with and some features that would be difficult to implement. One aspect of the MLP regressor that is disadvantageous for us is the difficulty in understanding its decision matrix (let's remember that these are the weights of just one perceptron out of 100 in the first layer). At least we get a partial confirmation that the food_expenses column is a very important feature.

Finally, before moving on, we should mention that no major complications have occurred in our programming work. This is because all the models are implemented in the same library (SciKit Learn). Therefore, the models have a similar syntax, similar usage and utilize a similar workflow. This allowed us to reuse most of our code during the evaluation of all models. In addition, we implemented our dummy classifiers similarly to SciKit Learn's implementation (mimicking their functions of 'fit' and 'predict'). This also allowed us to reuse the evaluation code for all other regressors.

3.3 Refinement

Having a better understanding of the feature importances, the successful initial models and the validity of our approach, we can now proceed and derive the final model. To do that, we will:

1. Select the model
2. Select 3 features based on the following criteria:
   - Feature weight
   - Feature applicability (how easy would be for Kiva to integrate such a question into their loan application form)
3. Retrain the model on the selected features.
4. Test the model performance on the test set.

We'll start first by selecting the model. Based on the performance table, derived above, it would seem reasonable to select the Decision Tree model. Another good reason for choosing it is that decision trees are easily explainable.

Next, we move on the the 3 features. Our choice is: food_expenses, farming_gardening_expenses and agricultural_household. We prefer the agricultural_household feature as it seems easier to apply in the field, however its weight is less than 1% of our model's prediction power, so we can drop it altogether.

We will use the grid search technique to optimize the hyperparameters of our decision tree model. We will use the same metrics and evaluate a grid of the following optionscagainst them:

hyperparameter	values
max depth	100, 200, 300, 400, 500
min leaf nodes	100, 200, 300, 400, 500
min samples leaf	2, 4, 6, 8, 10
min samples split	2, 4, 6, 8, 10

After the hyperparameter tuning, the resultant best model has the following hyperparameters:

hyperparameter	values
max depth	100
min leaf nodes	200
min samples leaf	2
min samples split	2

We will now assess the performance after retraining (we are expecting a slight drop in performance because we limited ourselves to 2 features):

metric	value
mean_squared_error_test	0.000123
r2_score_test	0.976352
mean_squared_log_error_test	0.000035
explained_variance_score_test	0.976353

4 Results

4.1 Model Evaluation and Validation

We can now combine all predictions and derive the poverty score from the predicted normalized income. In the future, as we add more countries to the estimation, our poverty score will more and more reflect global levels. Here is an initial distribution description of our poverty score:

metric	predicted income	poverty score
count	41544	41544
mean	0.890068	0.109932
std	0.071839	0.071839
min	0.103609	0.000511
25%	0.853509	0.056976
50%	0.903322	0.096678
75%	0.943024	0.146491
max	0.999489	0.896391

We will now join the region and house head female indicators from the original FIES data set. This will allow us to merge the poverty score with the Kiva loans data based on region and gender (as prescribed by the rules of the Kiva challenge). We will then provide visualizations for the results in the next chapter.

4.2 Justification

As we saw in the initial tests, the results of the dummy predictors (our chosen benchmarks) and our final model were (note that here the final model's metrics are on the test set):

regressor	mean squared error	r2 score	mean squared log error	explained variance score
dummy_0.0	0.797975	-160.057695	4.06E-01	0.00E+00
dummy_0.5	0.157458	-30.780255	0.054666	0.00E+00
dummy_1	0.016941	-2.419269	0.004708	-2.22E-16
final model on test set	0.000203	0.961063	0.000056	0.961064

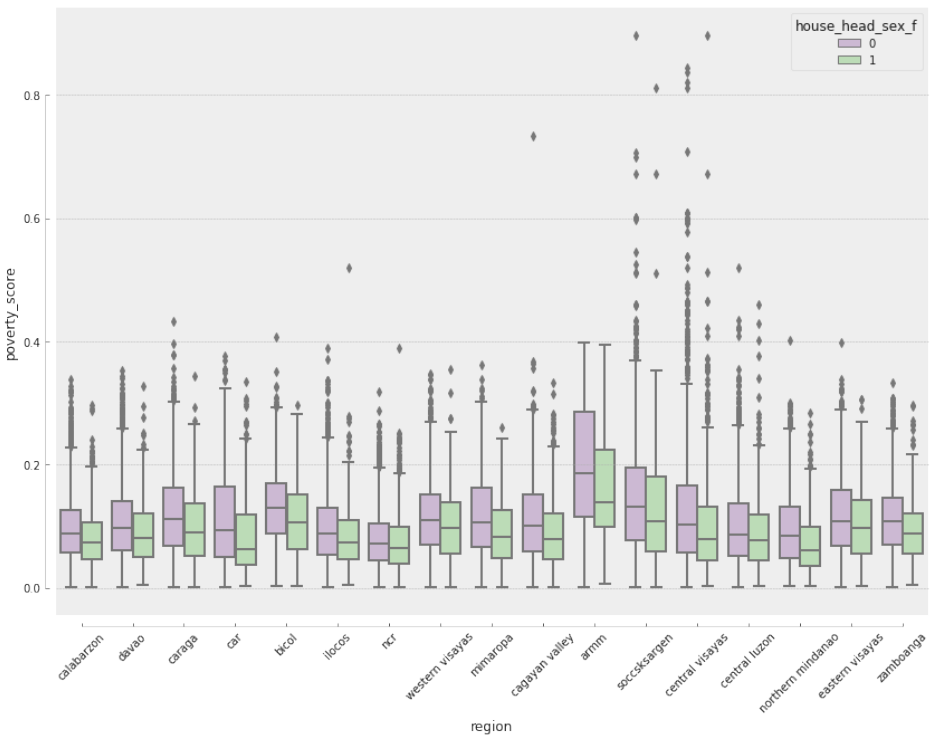
It seems that our final model generalizes well on unseen data.

5 Conclusion

5.1 Free-Form Visualization

Let us now visualize the distribution of the poverty score by region and gender:

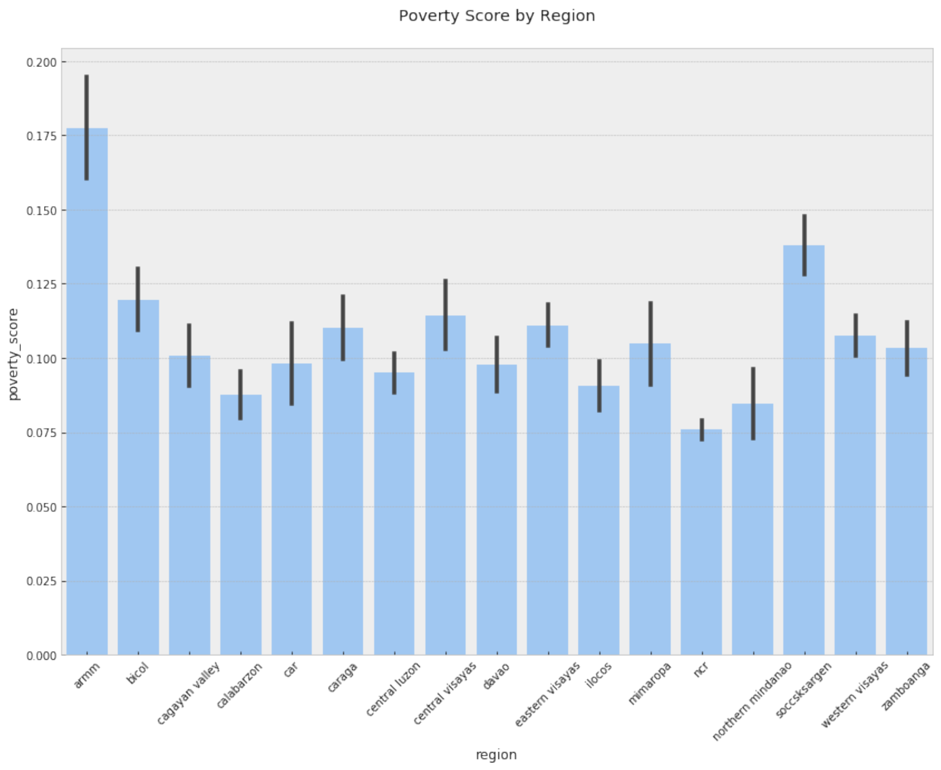
Figure 18: Poverty Score by Region and Gender



As we can see, the Visayas Island, the Soccsksargen province and the Administrative Region of Muslim Mindanao score consistently high in poverty score, compared to other regions. The interesting thing about Visayas is that, even though, the median, the 25th and 75th percentiles are within the country levels, there is a noticeable streak of outliers with extreme poverty scores.

We will now group the predictions by region and house head gender. In this grouping we'll take the mean value, although other measures could also be used. Another approach to the grouping could be to take the maximum poverty score for assessment of the worst affected areas.

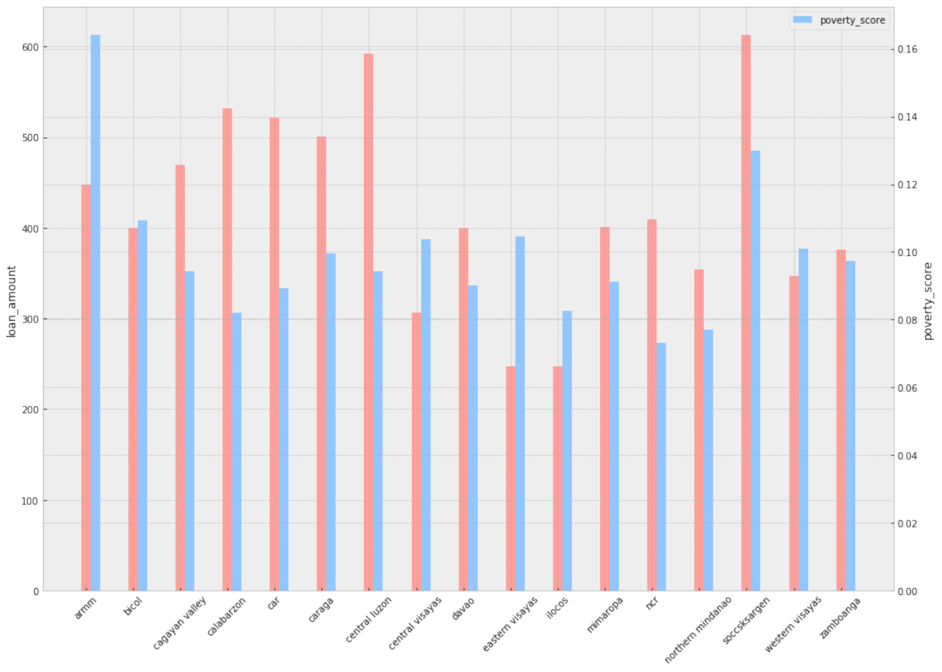
Figure 19: Poverty Score by Region



Our predictions about the poverty in the ARMM region are consistent with our initial observations about the poverty and unemployment levels in that region.

With the predictions of poverty grouped by gender and region in the FIES dataset, we can do a merge where we will join every mean grouped poverty score to every Kiva loan record by region and gender. We can now investigate the relationship between the granted loan amount and the corresponding poverty score for each region. This could help us identify areas where Kiva can invest more and have additional positive impact.

Figure 20: Mean Poverty Score and Mean Loan Amount by Region



The Kiva organization could increase its footprint in the ARMM and Eastern Visayas regions and thus contribute greatly to their improvement.

5.2 Reflection

The main reason for Kiva to create the poverty scoring challenge is to be able to better target underprivileged individuals. Thats why in this project, we decided to act as practically as possible and focus on the most important region for Kivas lending operation - the Philippines. Our challenge then was to define poverty and to find some data to help us derive it. For our poverty definition, we selected the lack of income or the inverse of the incomes normalized values for the country.

We were then faced with another challenge - to find data that contains both income and predictors of income. That data also had to cover all the regions of interest (the regions in the Philippines). We chose the Philippine Family Income and Expenditure Survey produced by the Philippine Statistical Authority.

The next step in our work was to transform the FIES data to a state usable by machine learning algorithms. That is, all numeric columns and one hot encoded categorical variables. We spent a large chunk of the project effort mapping, refining and transforming the FIES data set.

One of the serious challenges of the project was to map the loan location from the Kiva loans data set to regions in the Philippines using our naming convention. The reason for that was that the region field in the Kiva loans dataset actually contains free-form location description. In the easiest (and most prolific) case, the location description consisted of province name and city name separated by comma. In other cases, the description contained only a city name. Yet in other cases, the settlement name was that of a minor population center that we couldnt match to any major cities or provinces. Thus we ended with 150 000 successfully matched regions. Having the regions and genders allowed us to match the eventual poverty score derived from the FIES data set and join it to the Kiva loans data set.

The next task on our list was the development of the model itself. To that end we began by exploring the correlation between the income and other variables in the transformed FIES data set. We removed some other forms of income that had values close to our target variable. We then saw that several features from the FIES data set seemed to correlate highly with our income target. Those were: the food expenses, the farming expenses, the number of communication devices and the existence of some home appliances (refrigerator, AC, etc.).

We then normalized the data and passed it through several models and compared their performance to three dummy regressors. Our initial model choices were: AdaBoost regressor, Support Vector Machines regressor, Stochastic Gradient Descent regressor, Decision Tree regressor, Linear regression and Multi Level Perceptron regressor. Our dummy regressors were objects that predicted only (0, 0.5 and 1) respectively. We then recorded the results of all models on our selected evaluation metrics. Those metrics were: mean squared error, R2 score, mean squared log error and explained variance score.

We selected the 4 best performing models and made sure that their performance was better than that of the dummy regressors. We then derived the most important features for each of the models. We inspected the features and selected one model with its two top-performing features. Those were, respectively, the Decision Tree regressor with its food expenses and farming expenses features. Both features combined added to more than 90% of the feature weight. This allowed us to construct a lean, well explainable model that could be implemented in the field (e.g. by adding 2 simple questions to the loan application form).

We then derived the poverty metric out of the predicted income for all rows in the transformed FIES data set. We grouped the predictions for the entire dataset by region and gender and joined them to the Kiva loans data set. Thus we had a poverty metric for every loan record. In the end, we performed some visualizations of the results and investigated the poverty levels by region, gender, and compared them to the average granted loan amount per region.

5.3 Improvement

The Kiva organization needs a truly global model in order to be able to successfully assist underprivileged individuals around the World. Unfortunately, our current work only covers the Philippines. Even though that is the country, where Kiva grants most of its loans, the model could be much more global. To achieve that we will need to gather other household surveys and income data from other countries.

We can then transform them and find standard features available in all countries. We can then unite all the datasets and normalize the resultant data set again. This will give us a global measure of income and therefore - poverty.

Such a global solution will produce even better results for the poverty score since the normalized predicted income will be scaled across the world. This means that countries with populations having consistently low income will appear as poor compared to other, more economically-developed ones. This will give Kiva the opportunity to target poverty on any geographical scale: continent, country, region, province, etc. Of course this would entail currency conversion, standardization and many other challenges. However, a global poverty metric could dramatically improve Kivas positive impact on the World.