Feature Selection for Poverty Detection using Data Mining and Deep Learning

ROOBIKA V, Sri Eshwar College Of Engineering And Technology

Abstract - **Several countries in the world face the problem of Poverty. Different social organizations such as NGOs are working towards eliminating poverty by providing facilities such as financial aid, infrastructural facilities, basic primary education, and free health-care assistance. Ris study explored the ways to help different social organizations find the families that need social assistance and thereby help to reduce the poverty. Different data visualization tools were used to understand patterns and insights leading to the poverty. Several feature selection, data mining and deep learning techniques were utilized to find the household or families that would require social assistance. Publicly available dataset from Kaggle Website was used for this study. Kaggle is a data science website that contains open source datasets and provides a platform for Data Scientists to publish and showcase their work. A comparative analysis of different results accomplished was performed to derive conclusions.**

1 INTRODUCTION

Poverty prediction can help several social organizations working towards alleviating poverty to identify indigent families with that require assistance[1]. Social organizations such as NGOs can support such homeless and needy families with infrastructure facilities (houses), provide free primary education as well as free health assistance and welfare amenities. AI and machine learning can be used to predict poverty and help families vulnerable to extreme poverty with assistance and thereby, eliminate world poverty.

This study focuses on an extensive analysis of several poverty patterns using Data Science techniques in different regions of Costa Rica. This study can help Inter-American Development Bank, one of the organizations, in Latin America, working to identify and help thousands of families vulnerable to poverty with assistance and thus, reduce poverty [1].

The entire paper is divided into different sections. The section 2 is about literature review. Sections 3 and 4 describes the poverty dataset and different data cleaning techniques used in the independent study. Section 5 describes the data exploration techniques like data visualizing, cross-correlation and regression analysis. Section 6 discusses several feature reduction techniques like PCA and t-SNE. The next section 7 describes about different methods used for balancing the poverty dataset. Section 8, 9, and 10 talks about train-test splits, different data normalization techniques, and Grid Search. In the section 11 and 12 different supervised machine learning and deep learning models are implemented for poverty prediction. Section 13 evaluates the results of the models built on different versions of data. Section 14 implements sentiment analysis on poverty related tweets. Finally, the section 15 and 16 concludes the study and discusses the future scope.

2 LITERATURE REVIEW

Identifying which households are poor can be a tedious, expensive and time-consuming task as it requires an extended assessment of households by interviewing each household for several hours [19]. At times, it is difficult to obtain information on the income or expenditure of a family. Even household people may not be able to give this information due the lack of weekly records of different expenses and expenditure[19]. Alternatively, a Proxy Means Test is a technique which is used extensively. It helps to identity families that can be prone to poverty when income and other expense measures are not available [19]. The Proxy Means Test technique uses household level characteristic such as the material of walls, if a family owns a car or computer, sanitary conditions of household, and the number of people in the household as major indicators for predicting the poverty [19].

Kshirsagar et al. (2017) [23], used a statistical machine learning approach to improve the performance of Proxy Means Test tools for poverty prediction. They used [23] a logistic regression on a survey data obtained from Zambia. To improve the performance of the model and to reduce the over-fitting, the author used cross-validation and L2 regularization techniques. The results obtained were represented using box-plots. This technique helped different organizations to successfully predict poverty across both urban and rural locations in Zambia.

Natta Plulikov (2016)[30], suggested that household income levels or the amount of savings are not the only criteria for a family to be poor. Factors like education and household conditions have a very strong impact on the poverty conditions of different families. In this paper [30], the author used both supervised and unsupervised machine learning and deep learning techniques to predict poverty utilizing the data obtained from a survey of different Indonesian families. Since the data collected was categorical in nature, the author used multiple correspondence analysis as a feature selection technique. Multiple correspondence analysis is a feature reduction technique used specifically on categorical data [30]. The K-means clustering technique was adopted for finding the groups of poor and non-poor families. Different supervised learning techniques such as Decision Trees, Random Forests, and Binary Logistic Regression predicted the poverty levels with an accuracy of 85%. The accuracy of prediction was improved to 87% using artificial neural networks. Along with accuracy, sensitivity and specificity were taken into consideration for evaluating different machine learning and deep learning models.

McBride et al. (2015) [27], used different feature selection techniques for improving the performance of poverty prediction algorithm. The paper [27] used Principal Component Analysis (PCA) algorithm, in which important features that were useful in predicting the poverty (features showing most variance in the dataset) were selected. Regression Analysis based on R^2 maximization was also used for identifying the most important features for predicting poverty. Results obtained from PCA and regression analysis showed that the income levels, education levels and the number of people in the family are the most significant indicators of poverty. Based on the most important features selected, the paper [27] used Stochastic Regression approach such as quantile regression forest for predicting the poverty and bootstrap bagging to reduce the bias-variance trade-off [27]. The paper uses a t-test statistical technique for evaluating the performance of the model.

3 DATASET DESCRIPTION

The dataset for this independent study was obtained from the Kaggle website [1]. The dataset is collected by conducting a survey of Costa-Rican families in Latin America. The dataset mostly contains household level features such as education levels of the family members, details about materials from which houses are built such as wall, roof, and floors, details about sanitary conditions in household. In addition to this, information about source of cooking gas, source electricity and water in the household can be inferred from this dataset. Details about the number of computers, TVs, mobile phones, and number of tablets owned by family members are also revealed from this survey. Information on number of male, females and dependents in the households, yearly house rent details can also be inferred from this dataset. Majority of the features in the dataset were either binary (1/0) (already one-hot encoded) or numeric [1].

The detailed explanation about the features in the dataset can be found in the Table 1.

Table 1. Dataset Description for Poverty Prediction [1]

Attributes	Description
Id	Unique identifier for each family member
Target	Ordinal, poverty levels
idhogar	Unique identifier for each household
parentesco1	1/0 Head of household
parentesco2	1/0 Spouse/Partner
parentesco3	1/0 Son/Daughter
parentesco4	1/0 Stepson/Daughter
parentesco5	1/0 Son/Daughter in law
parentesco6	1/0 Grandson/Daughter
parentesco7	1/0 Mother/Father
parentesco8	1/0 Father/Mother in law
parentesco9	1/0 Brother/Sister
parentesco10	1/0 Brother/Sister in law
parentesco11	1/0 Other family member
parentesco12	1/0 Other non family member
v2a1	monthly rent
hacdor	1/0. Overcrowding by bedrooms
rooms	number of all rooms in the house
hacapo	1/0, Overcrowding by rooms
v14a	1/0, Household has bathroom
refrig	1/0, Household has refrigerator
v18q	1/0, owns a tablet
v18q1	number of tablets household owns
r4h1	Males younger than 12 years of age
r4h2	Males 12 years of age and older
r4h3	Total males in the household
r4m1	Females younger than 12 years of age
r4m2	Females 12 years of age and older
r4m3	Total females in the household
r4t1	persons younger than 12 years of age

Continued on next page

Table 1 – continued from previous page [1]

Attributes	Description
r4t2	persons 12 years of age and older
r4t3	Total persons in the household
tamhog	size of the household
tamviv	number of persons living in the household
escolari	years of schooling
rez_esc	Years behind in school
hhsize	household size
paredblolad	1/0 Outside wall is block or brick
paredzocalo	1/0 Outside wall is socket
paredpreb	1/0 Outside wall is cement
pareddes	1/0 Outside wall is waste material
paredmad	1/0 Outside wall is wood
paredzinc	1/0 Outside wall is zink
paredfibras	1/0 Outside wall is natural fibers
paredother	1/0 Outside wall is other
pisomoscer	1/0 Floor is mosaic/ceramic/terrazo
pisocemento	1/0 Floor is cement
pisoother	1/0 Floor is other
pisonatur	1/0 Floor is natural material
pisonotiene	1/0 No floor at the household
pisomadera	1/0 Floor is wood
techozinc	1/0 Roof is metal foil or zink
techoentrepiso	1/0 Roof is fiber cement, mezzanine
techocane	1/0 Roof is natural fibers
techootro	1/0 Roof is other
cielorazo	1/0 House has ceiling
abastaguadentro	1/0 Water provision inside dwelling
abastaguafuera	1/0 Water provision outside dwelling
abastaguano	1/0 No water provision
public	1/0 Electricity from CNFL, ICE, ESPH/JASEC
planpri	1/0 Electricity from private plant
noelec	1/0 No electricity in the dwelling
coopele	1/0 Electricity from cooperative
sanitario1	1/0 No toilet in the dwelling
sanitario2	1/0 Toilet connected to sewer or cesspool
sanitario3	1/0 Toilet connected to septic tank
sanitario5	1/0 Toilet connected to letrine
sanitario6	1/0 Toilet connected to other system
energcocinar1	1/0 No kitchen/no source of energy for cooking
energcocinar2	1/0 Energy used for cooking - electricity
energcocinar3	1/0 Energy used for cooking - gas
energcocinar4	1/0 Energy used for cooking - wood charcoal
elimbasu1	1/0 Disposal by tanker truck
elimbasu2	1/0 Disposal by botan hollow or buried

Continued on next page

Table 1 – continued from previous page [1]

Attributes	Description
elimbasu3	1/0 Disposal by burning
elimbasu4	1/0 Disposal by throwing in unoccupied space
elimbasu5	1/0 Disposal by throwing in river/sea
elimbasu6	1/0 Disposal mainly other
epared1	1/0 Walls are bad
epared2	1/0 Walls are regular
epared3	1/0 Walls are good
etecho1	1/0 Roof are bad
etecho2	1/0 Roof are regular
etecho3	1/0 Roof are good
eviv1	1/0 Floor are bad
eviv2	1/0 Floor are regular
eviv3	1/0 Floor are good
dis	1/0 Disable person
male	1/0 male
female	1/0 female
estadocivil1	1/0 Less than 10 years old
estadocivil2	1/0 Free or Coupled union
estadocivil3	1/0 Married
estadocivil4	1/0 divorced
estadocivil5	1/0 Separated
estadocivil6	1/0 Widow/er
estadocivil7	1/0 Single
hogar_nin	Number of children 0 to 19 in household
hogar_adul	Number of adults in household
hogar_mayor	of individuals 65+ in the household
hogar_total	of total individuals in the household
dependency	Number of dependents
edjefe	years of education of male head of household
edjefa	years of education of female head of household
meaneduc	avg. years of education for adults 18+
instlevel (1 to 9)	1/0, Education Levels
bedrooms	number of bedrooms
overcrowding	persons per room
tipovivi (1 to 5)	Rent Payment methods
computer	1/0 Household notebook/desktop computer
television	1/0 Household has TV
mobilephone	1/0 Household has Mobile phone
qmobilephone	Number of mobile phones
lugar1	1/0 Region Central
lugar2	1/0 Region Chorotega
lugar3	1/0 Region Pacific fico central
lugar4	1/0 Region Brunca
lugar5	1/0 Region Huetar Atlantica

Continued on next page

Table 1 – continued from previous page [1]

Attributes	Description
lugar6	1/0 Region Huetar Norte
area1	1/0, zona urbana
area2	1/0, zona rural
age	Age of house in years
SQBescolari	Escolari squared
SQBage	Age squared
SQBhogar_total	Hogar_total squared
SQBedjefe	Edjefe squared
SQBhogar_nin	Hogar_nin squared
SQBovercrowding	Overcrowding squared
SQBdependency	Dependency squared
SQBmeaned	Square of the mean years of education of adults
agesq	Age squared

4 DATA CLEANING

The dataset [1] was messy and needed lots of preprocessing. The first step toward cleaning the data was to invert the target variable. Initially, '1 : extreme poverty', '2 : moderate poverty', '3 : vulnerable'and '4 : non-vulnerable'. So for simplicity and for better understanding the data and correlations between different features and target variable, the Target feature was inverted. After inverting the Target variable became '0: non-vulnerable', '1: vulnerable', '2: moderate poverty'and '3: extreme poverty 'respectively.

```
#data types of different features
train_temp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9557 entries, 0 to 9556
Columns: 143 entries, Id to Target
dtypes: float64(8), int64(130), object(5)
memory usage: 10.4+ MB
```

Fig. 1. Total instances, aNributes, and data types of different features in the dataset [1]

Figure1 shows that 8 features were floats, 130 attributes were integer and 5 attributes were of object data type. Since some features in the dataset had Object as their data types but were actually numeric in nature. All such variables were converted from object data to float data.

$$train.dependency.astype(float) \tag{1}$$

Equation 1 shows that the data type of dependency attribute was change to float using "astype(float)" function.

According to the dataset description given in the table 1, all the members in the same family should have the same poverty level. The correct poverty level for a particular family is indicated by the poverty level of the 'parentesco1: head of the family'. Few of the families did not follow this relationship. Therefore, corrected the target variable for certain families.

Features like 'dependency' had incorrect data. Certain values were "yes" or "no" instead of being 1 and 0. So by intuition replaced the value "yes" with 1 and "no" with 0.

$$train.loc[(train.dependency == yes), dependency] = 1 \tag{2}$$

$$train.loc[(train.dependency == no), dependency] = 0 \tag{3}$$

Equations 2 [1] and 3 [1] shows that dependency value of "yes" was replaced with 1 and dependency value "no" was replaced with 0 respectively.

Dataset had lot of missing values denoted by 'NaNs'. 'NaNs' indicated that people did not answer all the questions during the poverty survey. Therefore, all the missing values were imputed. After performing an initial analysis 3 features : 'rez esc: Years behind in school', 'v18q1: no of tablets' and 'v2a1: monthly rent' had majority of missing values. Table 2 shows the count of missing values for these features.

After exploring the dataset, the feature 'v18q1: no of tablets' from the table 2 was found to be related to 'v18q: owns tablet' and all the rows having 0 for 'v18q: owns tablet' had 'NaN' for 'v18q1: no of tablets'. Therefore, by correlation, all the missing values in v18q1 were replaced with 0.

The feature 'v2a1: monthly rent' from the table 2 was found to be related to 'tipovivi1: owned and fully paid house'. All the rows having 'NaN' for monthly rent were fully owned houses. Therefore, all the missing values in v2a1 were replaced with 0.

Table 2. Features having majority of missing values [1]

Features	Count Missing values
rez_esc	7928
v18q1	7342
v2a1	6860
meaneduc	5
SQmeaned	5

Another feature 'rez_esc: Years behind in school' had missing values and was found to be related with age. All the people having rez_esc as 'NaN' were adults (having age $>$ 18 years). Hence, all the missing values in rez_esc were replaced with 0.

5 EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is a remarkably important step in Data Science. EDA is a process that helps in discovering patterns, outliers, and anomalies in the data. EDA can also help to test the several different assumptions and hypothesis, find and fill missing data without even building actual machine learning and deep learning models[6]. EDA also helps Data Scientists to determine the most significant attributes that can be used by the machine learning algorithms [6].

EDA can be performed using several techniques. Few of them include data visualization using bar charts, line charts, box plots etc., cross-correlation analysis and regression analysis for determining the most important features[6].

5.1 Data Visualization - Finding insights from the data

The data visualization was performed to discover patterns and trends related to poverty. Utilized the Matplotlib[2] and Seaborn [38] data visualization package for Python for visualizing the data.

Figure 2 shows the distribution of the Target variable (poverty distribution). Figure 2 indicated that a large number of families belong to the class of non-vulnerable whereas only few families belong to the class of vulnerable, moderate, and extreme poverty. Figure 2 also showed that there is an imbalance in the distribution of the class variable. Therefore, the target class variable has to be balanced before applying any machine learning models (to avoid machine models being erratic and biased). The balancing of the target variable is performed in the section 7.

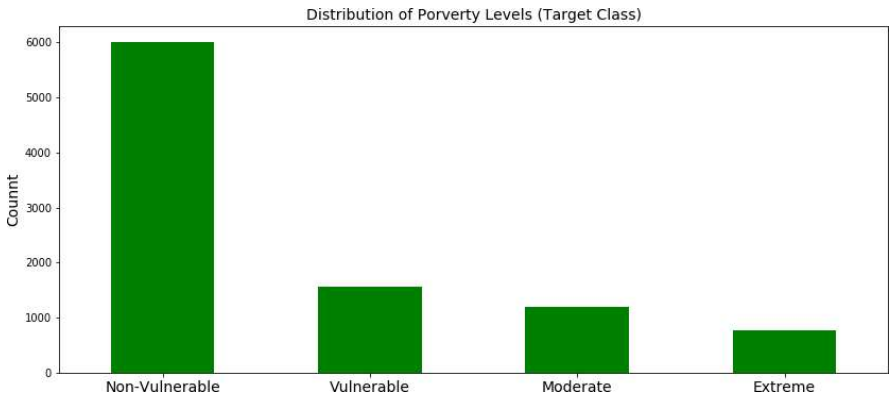

Fig. 2. Distribution of Poverty Levels (Target Variable)

Figure 3 shows the quality of the housing material such as floors, roofs, and walls for the families belonging to the various poverty levels. The quality of the housing material varies from 0 to 6 where 0 means the worst quality and 6 means the best quality of the housing material. Figure 3 also points that approximately 80% of the families that are not vulnerable to poverty have better quality houses as indicated by the overall quality of 3, 4, 5, and 6 whereas the majority of the families that are vulnerable to poverty have poor quality homes as shown by the overall quality of 0,1, and 2. Therefore, if households are not vulnerable to poverty, they have better quality houses otherwise households belonging to the category of extreme and moderate poverty have poor quality houses.

Figure 4 shows the levels of education for the families belonging to different poverty levels. The education levels described in the figure 4 (lower to higher levels) are No Education, Incomplete

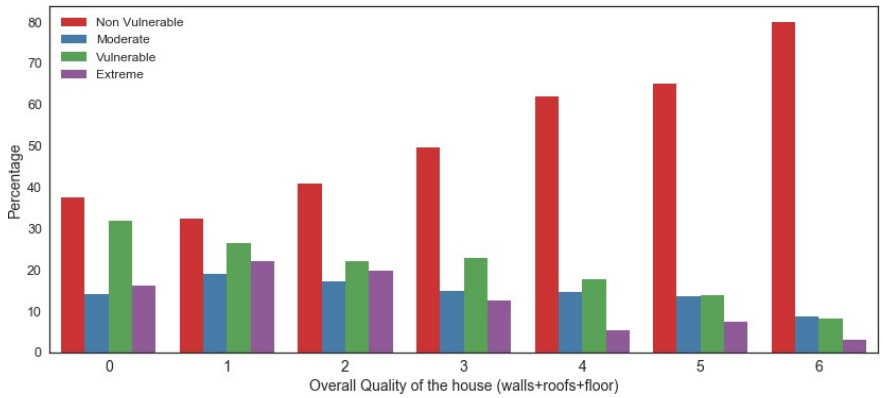

Fig. 3. ality of Housing materials (such as roofs, floors, and walls) for different Poverty Levels. Families having beNer quality houses, face liNle prone to poverty [21]

Primary, Complete Primary, Incomplete Secondary, Complete Secondary, Incomplete High school, Complete High school, Undergraduate, and Graduate. Figure 4 indicates that the family members completing at least High school education or higher are less vulnerable to poverty. Figure 4 also explains that approximately 20% of the families having very limited or No Education belong to the category of extreme poverty. Also, 35% of the families that have completed only primary level of education are vulnerable to poverty. Therefore, if family members are more educated the chances for that family to be prone to poverty are reduced. Lower levels of education signify more poverty levels.

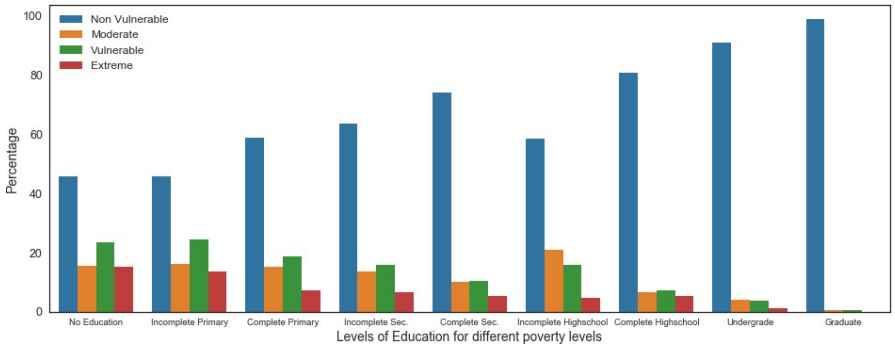

Fig. 4. Levels of Education for different Poverty Levels. Families having more educated people are less prone to poverty [21]

Figure 5 is a box plot showing the effect of average years of education on the poverty. The box plot shows that if members in the family are more educated (i.e. have high average years of education), then those families are less vulnerable to the poverty. Figure 5 indicated that family

members having maximum of 18 years of education are less prone to poverty whereas family members having 9-13 years of education are more vulnerable to poverty.

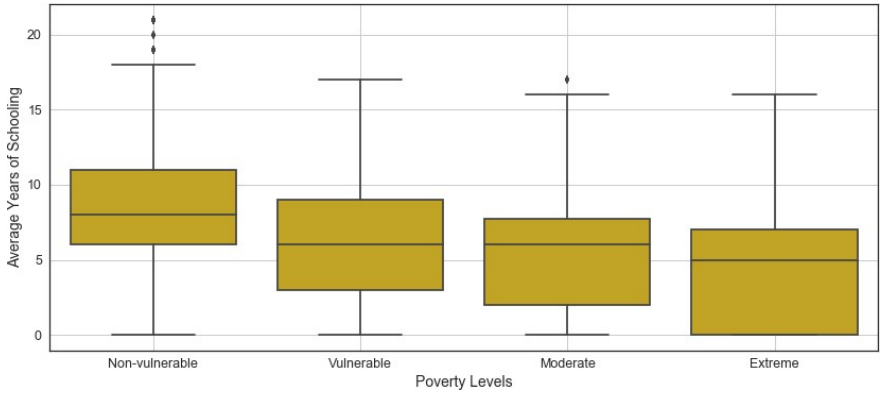

Fig. 5. Effect of Average Years of Education on Poverty Levels. Families with grater years of average education show reduced poverty levels[21]

Figure 6 shows the consequence of the number of children (dependents) in the age group from 0 to 19 years on the poverty levels. Approximately 25% of the families having 0 children are not vulnerable to poverty. Figure 6 also reveals that families having either 1,2 and 3 number of children are less vulnerable to poverty, whereas families having more than 3 children are extremely prone to poverty. Therefore, this indicates that more the dependents in the family, more chances are for that family to be affected by poverty.

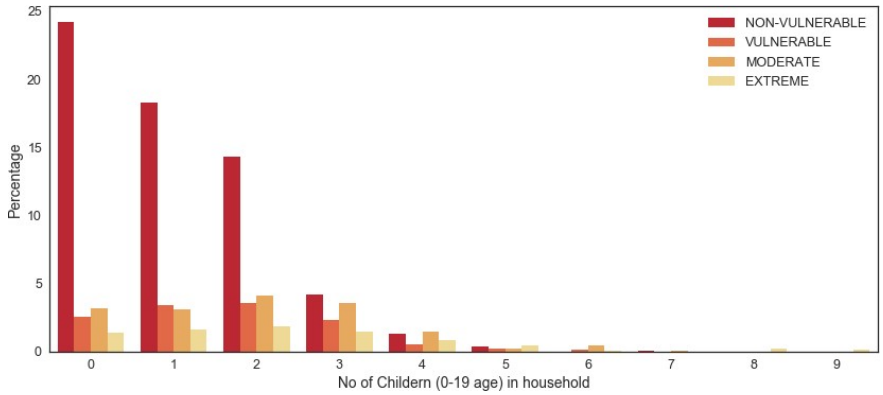

Fig. 6. Effect of number of Children in the household on the Poverty. Family is more vulnerable to poverty if there are more dependents (children) living in the house [21]

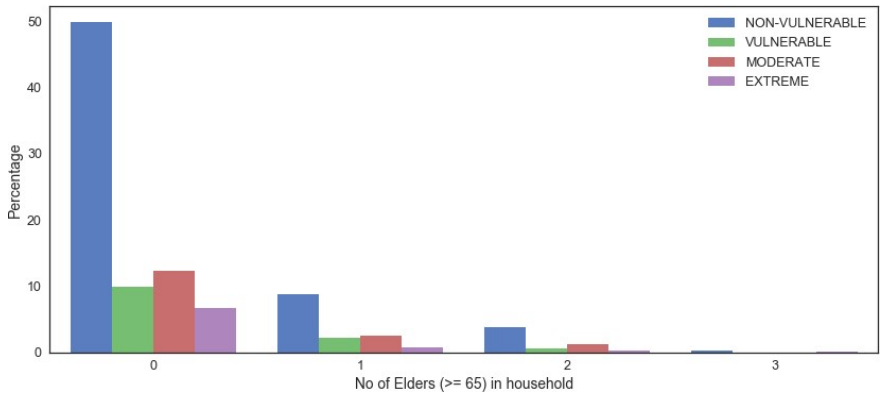

Fig. 7. Effect of number of Elders in the household on the Poverty. Household is more vulnerable to poverty if there are 1 or more elders in the family[21]

Similarly, figure 7 shows the effect of the number of elders ≥ 65 years of age (dependents) on the poverty levels. This also indicates that the more dependents i.e. the Elder members in the family, greater the chances for that family to be prone to the poverty conditions. Figure 7 shows that around 50% of the households having 0 elders in the families are not vulnerable to the poverty.

Figure 8 shows the effect of the number of adults (non-dependents) on the poverty levels. Figure 8 depicts that if the number of adults in the households is less (e.g., 1, 2, 3 or 4), then these households are more vulnerable to severe poverty conditions whereas households having more adults (≥ 4) are less prone to poverty conditions.

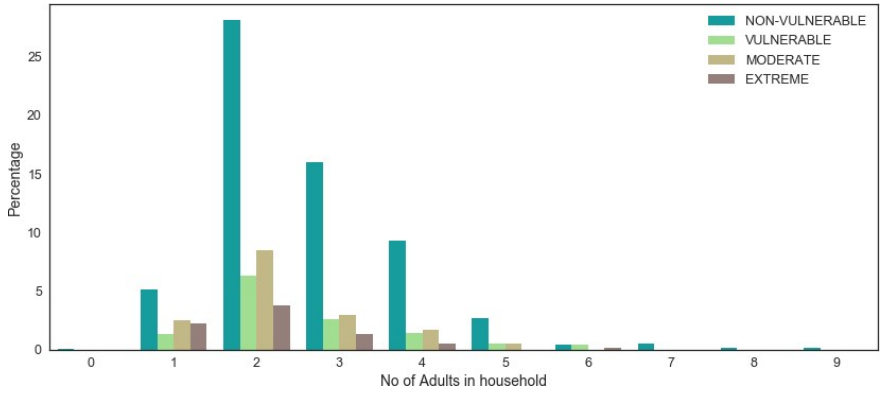

Fig. 8. Effect of number of Adults in the household on the Poverty. Household is less vulnerable to poverty if there are more adults in the family [21]

Figure 9 is a box plot which shows the effect of number of males in the household on the Poverty Levels. Since, males are responsible for the financial stability in the family, households having more

males tend to show fewer poverty patterns. Figure 9 shows this trend where households having ≥ 3 males are less vulnerable to poverty whereas households having $\prec 3$ males are more vulnerable to poverty.

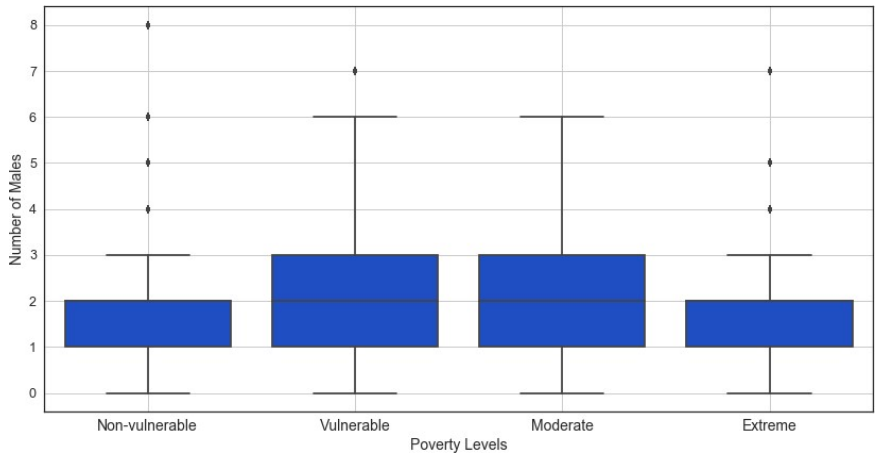

Fig. 9. Number of Males in the household and its effect on Poverty Levels. Household is less prone to poverty if there are more than 2 males in the family [21]

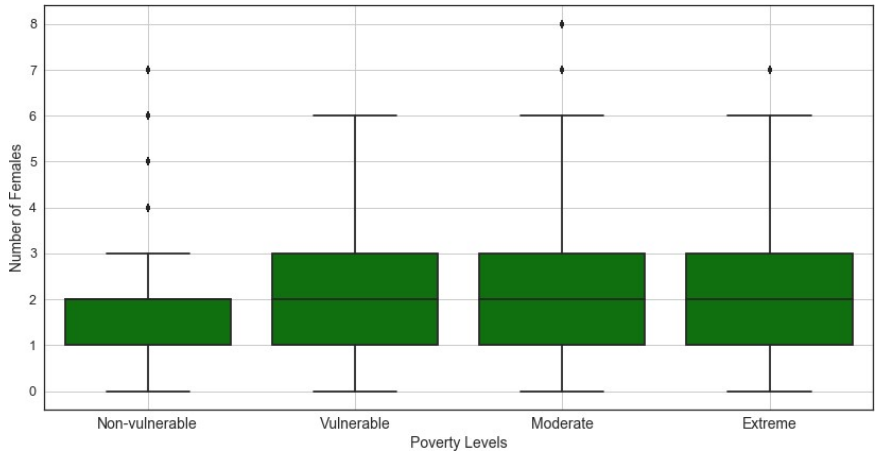

Fig. 10. Number of Females in the household and its effect on Poverty Levels. Household is less prone to poverty if there are fewer females (dependents) in the family [21]

Similarly, figure 10 is a box plot which shows the effect of number of females in the household on the Poverty Levels. In most of the families, females are considered to be the dependent on the

household men and contribute lesser to the financial stability of the house. Therefore, households having more number of females tend to show more poverty patterns. Figure 10 shows this trend where households having ≥ 2 females are more vulnerable to poverty whereas households having $\prec 2$ females are less prone to poverty.

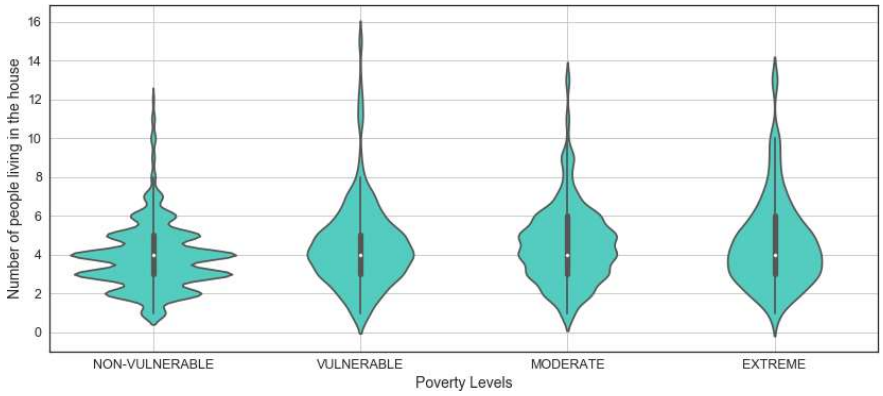

Fig. 11. Effect of number of persons living in the house on the Poverty Levels. Family is vulnerable to poverty if number of people living in the house are more than 5 [21]

Figure 11 is a violin plot which indicates the effect of the number of people staying in the house on the poverty levels. From the violin plot, it can be concluded that a family is vulnerable to poverty if there are over 8 family members living in the household. If a household has around 4 to 5 family members living in the house, then these households are less likely to suffer severe poverty situations.

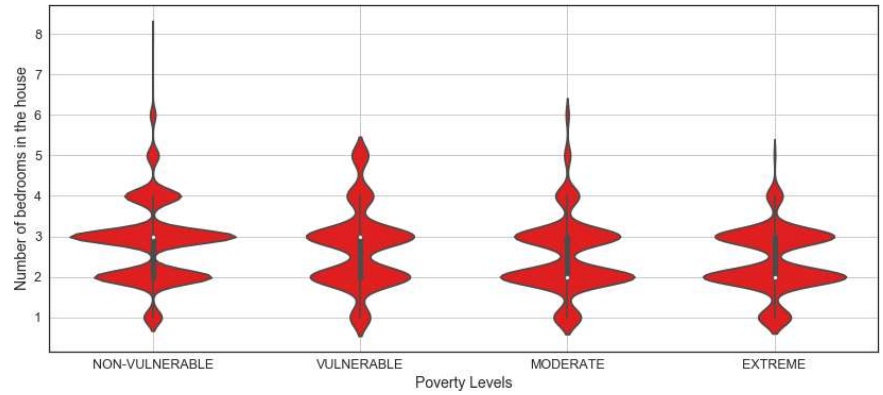

Fig. 12. Effect of number of bedrooms in the house on the Poverty Levels. Families are not prone to poverty if they live in a bigger house [21]

Figure 12 is a violin plot which explains the effect of the number of bedrooms in the house on different poverty levels. From the violin plot, it can be inferred that if families living in the house have at least 2 bedrooms per house, then these families are not considered to be in poverty. Whereas, families having 2 bedrooms in their houses can be considered vulnerable to poverty.

Figures 13, 14, 15, and 16 shows the effect of number rooms in the house and number of people living in the house on the poverty levels.

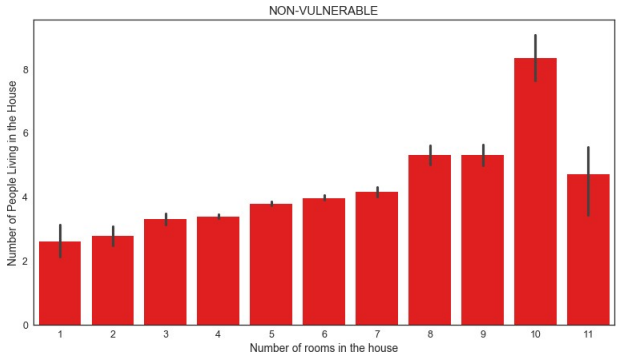

Fig. 13. Relationship between number of rooms and number of people staying in the house for non-vulnerable poverty conditions [21]

Figure 13 shows the relationship between the number of rooms in the house and the number of people staying in the house for non-vulnerable poverty conditions. Figure 13 implies the number of people living in the house is quite less as compared to the number of total rooms in the household. On an average around 3 to 5 people staying in the house having about 5 to 6 rooms in total are not vulnerable to poverty.

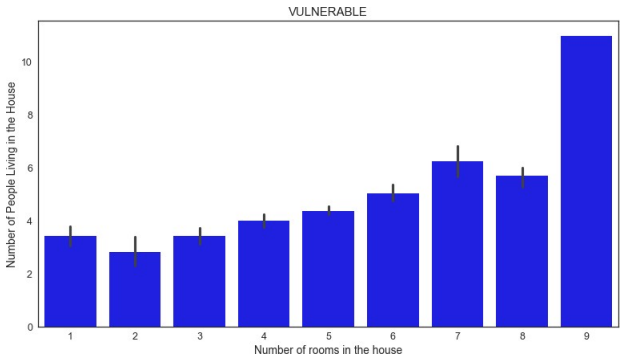

Fig. 14. Relationship between number of rooms and number of people staying in the house for Vulnerable poverty conditions [21]

Figure 14 shows the relationship between the number of rooms in the house and the number of people residing in the house for vulnerable poverty conditions. Figure 14 infers the number of

people living in the house is quite more as compared to the people living in non-vulnerable poverty conditions. On an average around 3 to 5 people staying in the house having about 2 to 4 rooms in total are likely to be vulnerable to poverty.

Figure 15 and 16 shows the relationship between the number of rooms in the house and the number of people staying in the house for moderate and extreme poverty conditions receptively. Figure 15 and 16 infers that the number of people living in the house is always more as compared to the number of rooms (i.e. size of the household). Figure 16 shows that on an average around 5 to 6 people stay in the house having just 1 or 2 bedrooms and thus show extreme poverty conditions.

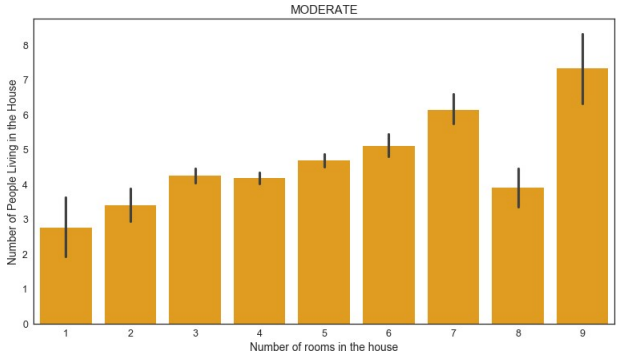

Fig. 15. Relationship between number of rooms and number of people staying in the house for Moderate poverty conditions[21]

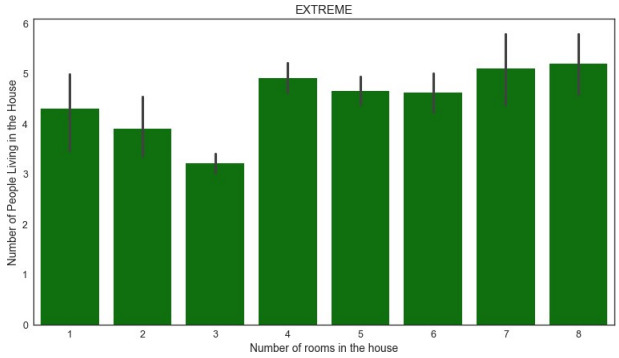

Fig. 16. Relationship between number of rooms and number of people staying in the house for Extreme poverty conditions [21]

Figure 17 shows the effect of the number of Tablets each family members owns on different poverty levels. Figure 17 indicates that if each of the family members owns a tablet than those families are not vulnerable to Poverty. For example, the families having around 5 to 6 people living in the house owns at least 5 to 6 tables and are, therefore, not vulnerable to poverty. Whereas, when the number of people living in the household is more than the number of tablets, then these families

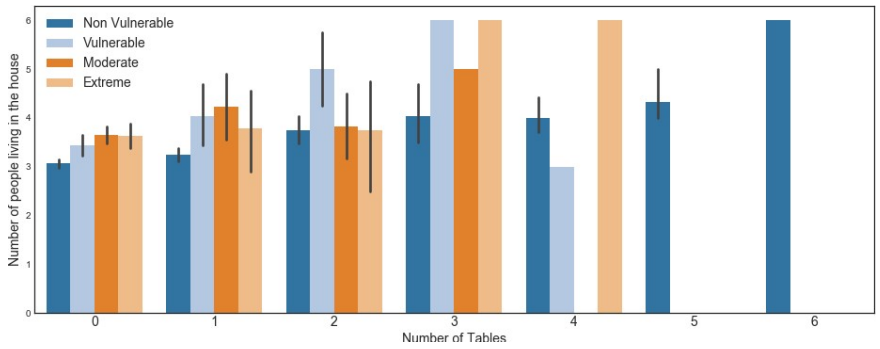

Fig. 17. Effect of number of Tablets each family members owns on different poverty levels. Families do not suffer from poverty if atleast there is one table/computer in the family [21]

are either moderately poor or suffer from extreme poverty conditions. For example, households having sizes of around 4 to 5 people owns only 2 or fewer tablets and suffer from extreme poverty.

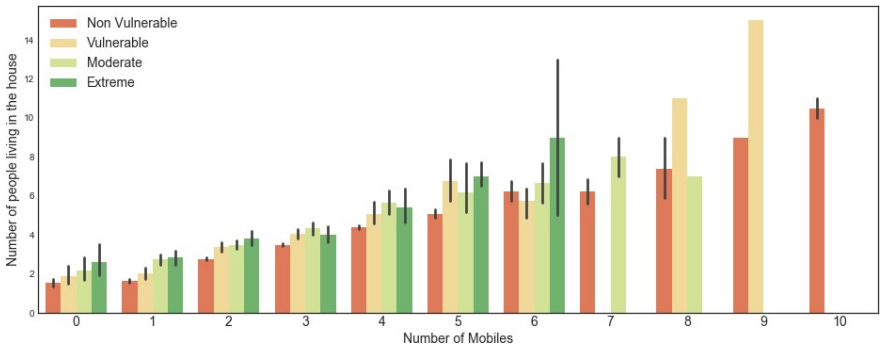

Fig. 18. Effect of number of Mobile phones each family members owns on different poverty levels. Families do not suffer from poverty if all the members in the family owns a mobile phone [21]

Figure 18 describes the effect of the number of mobile phones each family members owns on different poverty levels. This indicates that more the number of mobile phones a household owns lesser are the chances for that family to be considered as vulnerable to poverty conditions. For example, in figure 18 families having $\leq$2 mobile phones suffers from either extreme or moderate poverty condition.

Figure 19 shows the effect of yearly rent of the house and people living in the house on the non-vulnerable category of poverty levels. It indicates that the people with non-vulnerable poverty levels either pay 0 rent (i.e. houses are fully owned by the family) or pay rent$\geq$500000 per year. It can also be concluded that since it is possible for the families to pay such huge amount of rent, these families are not prone to poverty. Figure 19 shows that there is one outlier having rent around $\geq$ 2000000 in the data for non-vulnerable poverty category.

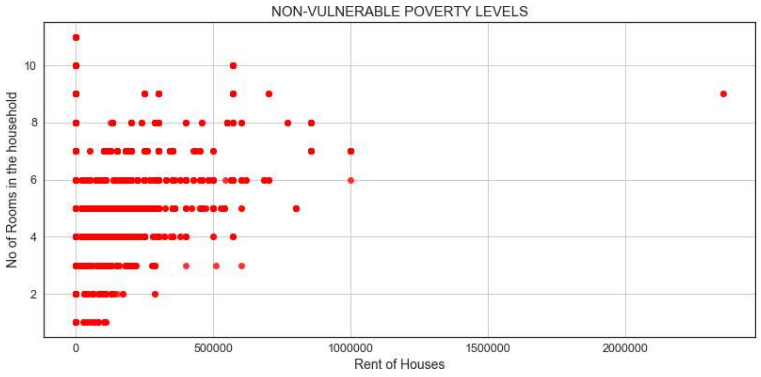

Fig. 19. Effect of Rent of the house and people living in the house on Non-Vulnerable Poverty Levels. Families not vulnerable to poverty either have a their own house or have the ability to pay higher rents than the families prone to poverty [21]

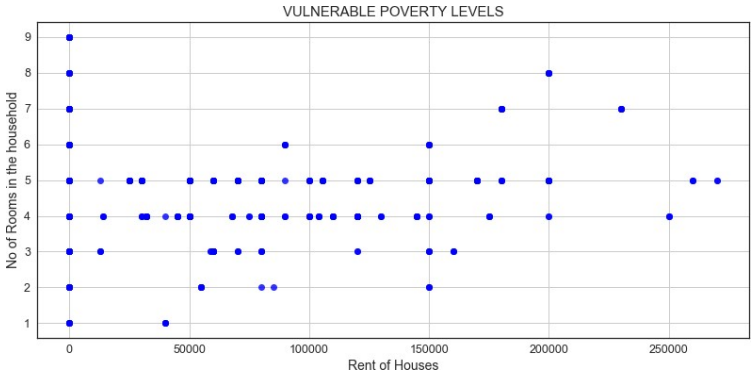

Fig. 20. Effect of Rent of the house and people living in the house on Vulnerable Poverty Levels. Families vulnerable to poverty tends to pay lesser rent than the families not prone to poverty [21]

Figure 20 indicates the effect of yearly rent of the house and people living in the house on the vulnerable category of poverty levels. It shows that most of the families belonging to vulnerable poverty levels pay yearly rent around 50000 and 150000. Also, the families belonging to this category of poverty levels pays comparatively lesser amount of rent than the families that are not prone to poverty. Additionally, very few families pay rent of 0 (i.e. very few households have their own houses).

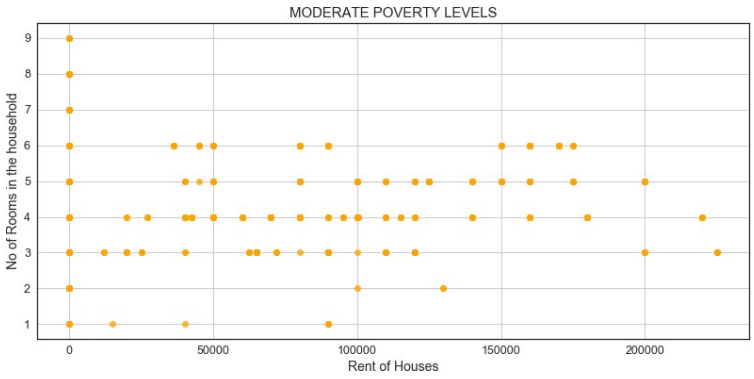

Fig. 21. Effect of Rent of the house and people living in the house on Moderate Poverty Levels. Families showing moderate poverty levels cannot afford to stay in luxury houses and pay less rent as compared to the families not prone to poverty [21]

Figure 21 indicates the effect of yearly rent of the house and people living in the house on the moderate category of poverty levels. It shows that most of the families belonging to moderate poverty levels pay yearly rent around 50000 and 100000. Therefore, it can be concluded that families in the moderate poverty levels cannot afford to live in the luxury houses having huge yearly rents.

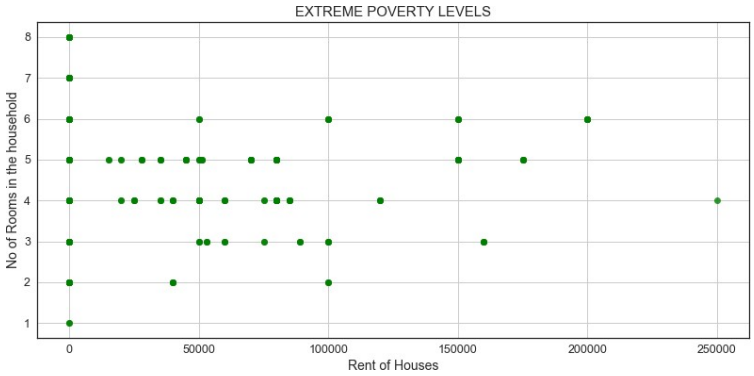

Fig. 22. Effect of Rent of the house and people living in the house on Extreme Poverty Levels. Families showing extreme poverty conditions also cannot afford to stay in luxury houses and pay less rent as compared to the families not prone to poverty [21]

Figure 22 indicates the effect of yearly rent of the house and people living in the house on the extreme category of poverty levels. It shows that most of the families facing extreme levels of poverty pay yearly rent ≤0000. Therefore, it can be concluded that families in the extreme poverty conditions cannot afford to stay in expensive houses.

5.2 Correlation Analysis

Cross-correlation analysis determines the degree of relationship between the variables and the direction of its variation [11]. The correlation value ranges from -1 to + 1 where the sign depicts the direction of correlation. The correlation is said to be negative (indicated by - sign), if one of the variable increases while the other variable decreases. The correlation is said to be positive (indicated by the + sign) if both variables increase or decrease at the same time. Correlation is 0 if there exists no relationship between the two variables [11]. The formula to calculate the cross correlation is given in the equation 4 [11] where S_x and S_y denotes the standard deviation for two variables X and Y, while $(X - \bar{X})$ and $(Y - \bar{Y})$ denotes the covariance between the two variables X and Y

$$P(X,Y) = \frac{1}{n-1} \quad \frac{(X-\bar{X})(Y-\bar{Y})}{S \qquad S} \tag{4}$$

Correlation analysis helps to understand how various features in the dataset are related to the target variable. It also helps to understand which features in the dataset are redundant and can be eliminated. This knowledge can be used to select the best features for predicting poverty conditions.

The heat-maps can be utilized to visualize the correlations between the target variable and other independent variables. The advantage of using heat-maps is that it is a great visualization tool and easily helps to draw conclusions from the given dataset.

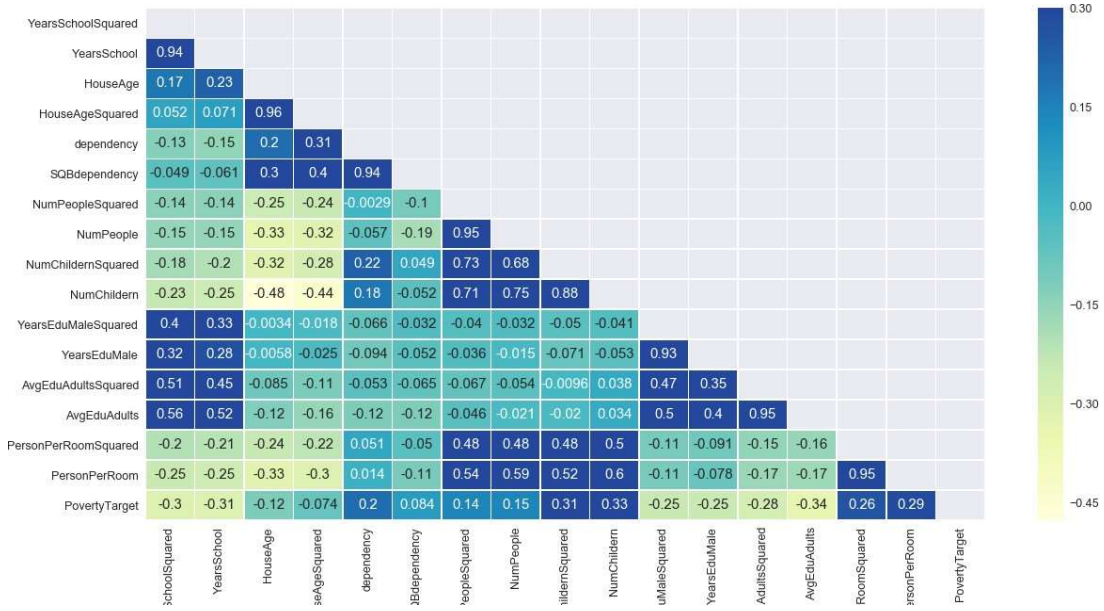

Fig. 23. Cross Correlation of all the Squared Variables with the Target variable. All the Squared Variables are highly correlated with the non-squared variables and are redundant [21]

It can be inferred from the figure 23 that the certain variables which are squares of other variables are highly correlated. This indicates that one of them is redundant and can be eliminated. For

example, the 'YearsSchoolSquared' variable is squared of 'YearsSchool' variable and contributes to 0.94 of correlation. So this indicated the redundancy and therefore, either one of the feature can be eliminated. Similar conclusions can be inferred from other squared variables.

Using figure 23 for analysis, poverty value of 0 indicated no poverty and poverty value of 3 indicated extreme poverty and values in between 0 to 3 indicated moderate poverty. From figure 23, the feature 'dependency' shows a positive correlation with the target 'poverty' variable. Thus, concluded that the families with more dependents are more vulnerable to poverty. Also, the heat-map showed that if men in the family are more educated then those households are less prone to poverty (shows a negative correlation). Also, households consisting of more people are poorer as indicated by a positive correlation in the heat-map. Correlation analysis also showed that older the house, the less probability of a particular family to be vulnerable to poverty.

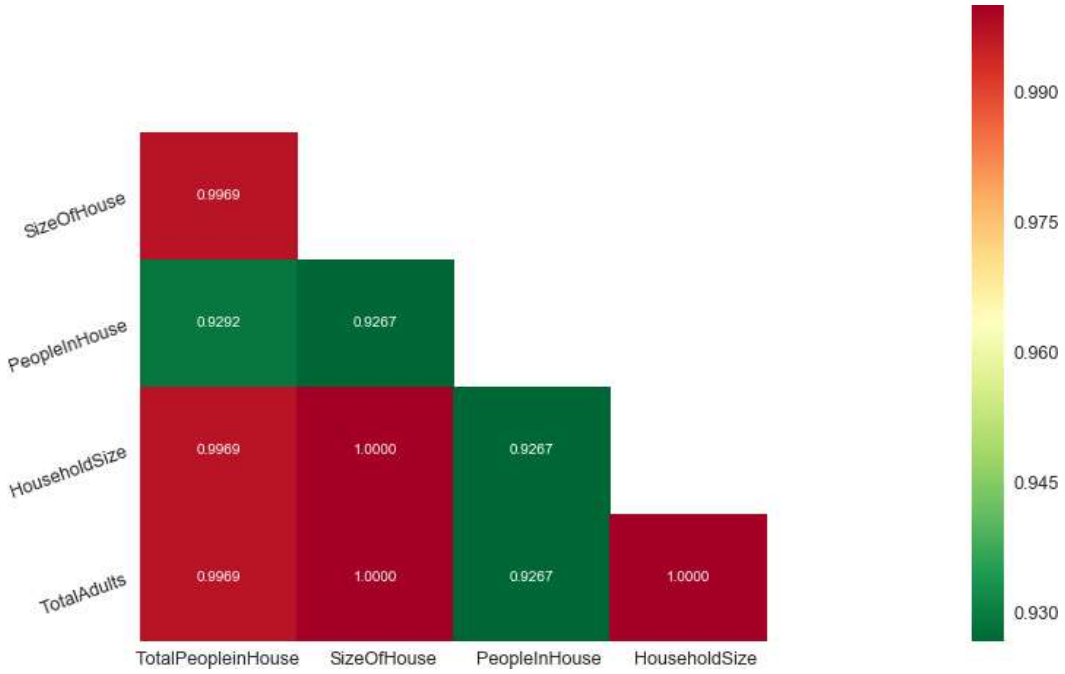

Fig. 24. Features having more than 90% Cross-Correlation with the Target class[21]

Figure 24 shows that feature 'HouseholdSize'has a correlation of 1 with features 'SizeofHouse'and 'TotalAdults '. Using this correlation information, features 'SizeofHouse 'and 'TotalAdults 'can be dropped. Similarly, 'TotalPeopleinHouse 'has 0.99 correlation with 'HouseholdSize '. Hence, 'TotalPeopleinHouse 'can be removed from the dataset.

From figure 25, it can be concluded that worst the 'Housing Material 'used more vulnerable a family would be to the poverty. Similarly, if there are no or very few 'Facilities 'such as electricity,

Fig. 25. Cross Correlation Analysis of subset of features with the Target variable [21]

water, sanitary needs in the house, more prone that family would be to poverty. Also, if there is more 'Dependency 'or more 'Overcrowding'of people in the house, then the households would be more vulnerable to poverty.

5.3 Regression Analysis

Regression Analysis is another method that can assist Data Scientists to explore patterns and trends in the given dataset. It is often considered to be a part of Exploratory Data Analysis (EDA).

A regression analysis is a type of a predictive method in machine learning. It attempts to find an association between a target variable, also known as the dependent variable and one or more independent variables, also known as explanatory variables [31]. The target variable is usually a variable on the Y-axis whereas the independent variable is a variable on the X-axis. Regression analysis works by fitting a curve (either a parabola or hyperbola) or a straight line to the independent and dependent variables The main aim is to reduce the gap between the data and the fitted straight line or the curve [31].

The foremost advantage of using regression analysis is that it examines if there is a significant relationship between the target and exploratory variables. It also determines the degree to which the exploratory variables changes the target variable [31].

Both linear and polynomial regression were explored as a part of this independent study. The principal difference between the two is that linear regression attempts to fit a straight line whereas a polynomial regression seeks to fit a curve (to discover more complex relationships between the data points) [31].

The R^2 method was utilized for evaluating the model. R^2 contains a baseline model which always outputs the average value of the target variable [13]. Any regression model that is developed is always compared with the R^2 model. The value of R^2 lies in the range from 0 to 1 where the value = 0 indicates that the built model is equal to the base model whereas the value = 1 reports that the model built is the best model anyone can built [13]. The equation of R^2 is mentioned in the equation 5 [31].

$$R^2 = 1 - \frac{SSE}{SST} \tag{5}$$

where, in equation 6 [13], SSE = Sum of Squared Errors for regression model.

$$SSE = \sum_{1}^{n} (y_i - \hat{yi}) \tag{6}$$

and, in equation 7 [13] SST = Sum of Squared Errors for baseline model.

$$SST = \sum_{1}^{n} (y_i - \bar{yi}) \tag{7}$$

Regression Analysis was performed using Scikit Learn [29] machine learning library for python. Visualizations were made using Matplotlib [2]. For analysis, poverty i.e. the Target variable was used as a dependent variable and other features such as “no of people living in the house”, “no of children in the age of 0-19”, “no of adults”, “years of schooling” and “no of females in the house” as independent variables. From the dataset, poverty level 0 indicated no poverty whereas poverty level 3 indicated extreme poverty.

Figure 26 shows the number of people in a given family is directly related to the size of the family (i.e., the number of people living in the household) and have a linear and positive relationship.

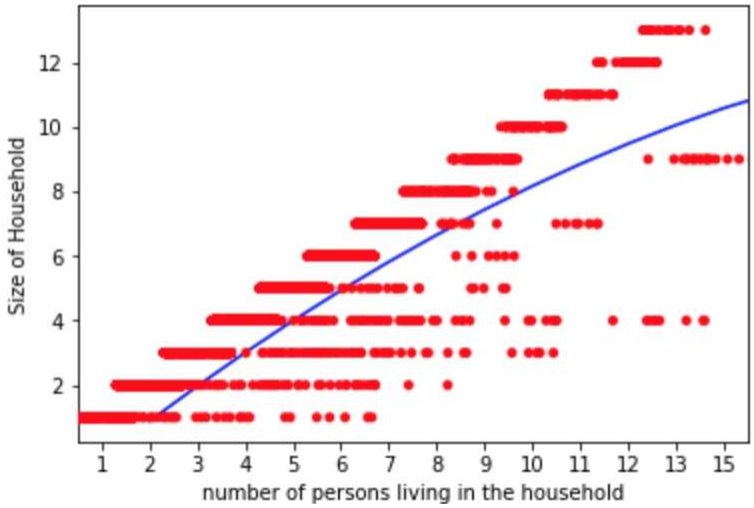

Fig. 26. Regression Analysis - Number of persons living in the house vs size of the house.

Figure 27 revealed that more children in the family (more dependents) indicates more poverty and shows a positive relationship with the target variable. Thus, as the number of children in the family increases, the probability of a family being prone to poverty also increases. The regression line is not linear but still shows trend in the data.

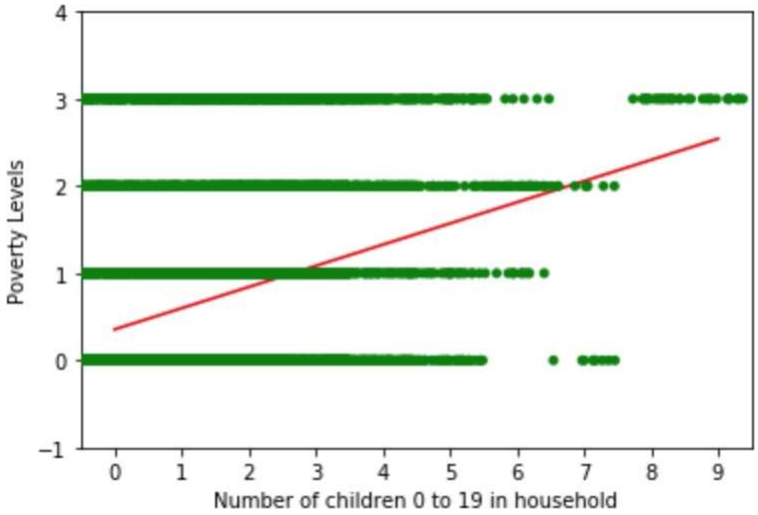

Fig. 27. Regression Analysis - Number of children (0-19) in the household vs different poverty levels. Shows a positive relationship indicating that families having more children are prone to suffer from poverty conditions

Figure 28 concluded that if there are more adults in the family than those families are less likely to be poor, thus, showing a negative relationship with the Target variable. Thus, if the total number of adults in the family increases, then that family is considered to be less prone to poverty. The regression line is not linear but still shows trend in the data.

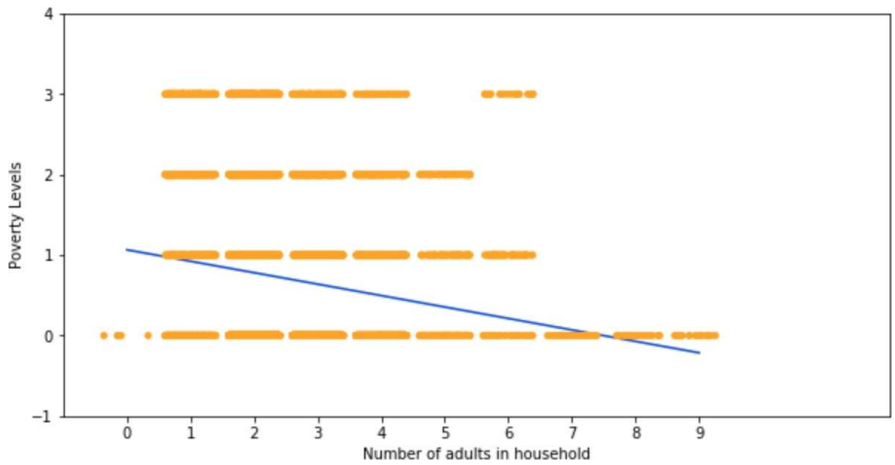

Fig. 28. Regression Analysis - Number of adults in the household vs different poverty levels. Shows a negative trend indicating families having more adults are not prone to poverty

Similarly, figure 29 showed that the households having more educated people are less inclined towards poverty conditions. Thus, shows a negative relationship with the different poverty levels. The regression line is not linear but still shows trend in the data.

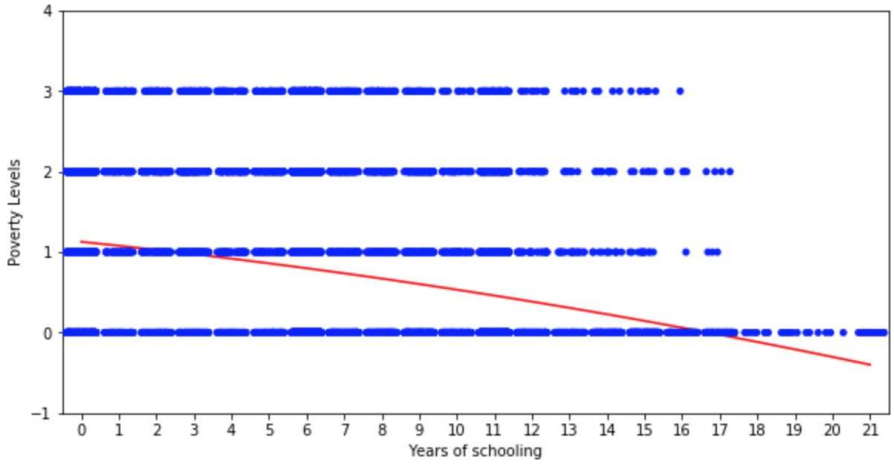

Fig. 29. Regression Analysis - Years of schooling vs different poverty levels. Shows a downward trend indicating education levels has a effect on poverty.

Figure 30 illustrated that families having more females (usually being dependent on men) are more liable to poverty situations. They show a positive relationship with different poverty levels. The regression line is not linear but still shows trend in the data.

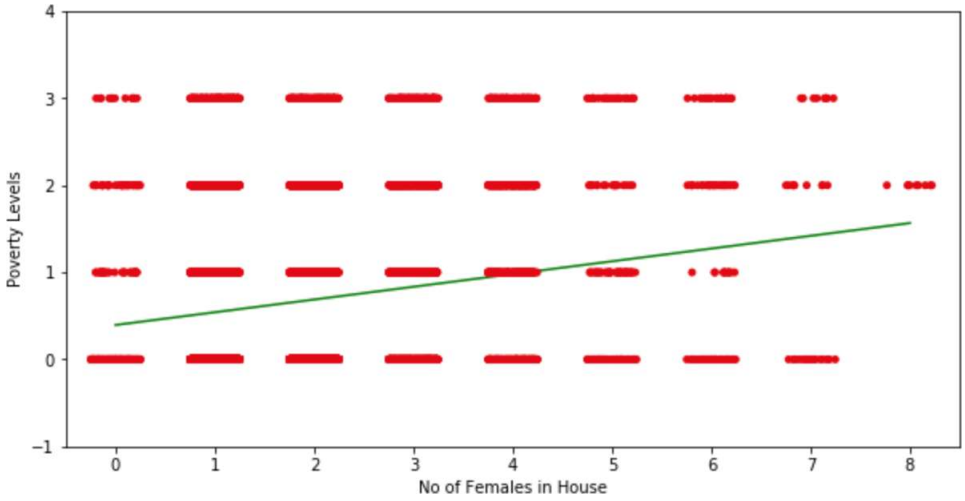

Fig. 30. Regression Analysis - Number of females in the household vs different poverty levels. Shows a positive trend between poverty levels and number of females in the house.

6 FEATURE REDUCTION METHODS

Different types of dimensionality reduction and feature selection techniques were explored and implemented as a part of this independent study and are discussed in the sections 6.1 and 6.2.

6.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique that reduces the number of features from a very high dimensional dataset. The reduced set of features still maintains the maximum amount of variance as compared to the high dimensional data [8]. Thus, PCA can helps to overcome the problem of the curse of dimensionality by removing all the less importance and redundant features from the dataset.

PCA is a linear and mathematical algorithm [12]. N different principal components are constructed by converting a large number of correlated features in the dataset into an N number of uncorrelated features [8]. All the principal components are always perpendicular to each other i.e., all the principal components are independent of each other[8]. The first principal component gives maximum variance in the data whereas all the other principal components gives the remaining variance in the dataset [8]. The data is projected on these principal components.

There are two ways to use PCA. First, tell PCA algorithm to reduce the dataset to n features such that the data retains 95 % of the total variance. In second method, tell the PCA algorithm to reduce the data to n features and then calculate the total variance retained by the reduced dataset [12].

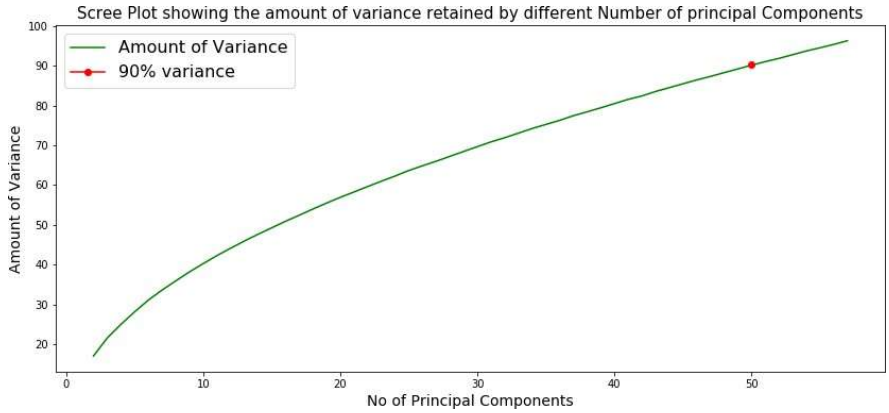

Fig. 31. Amount of Variance retained by different numbers of Principal Components of PCA algorithm for poverty dataset

Figure 31 is a Scree plot [8] showing the amount of Variance retained by different numbers of Principal Components of PCA algorithm for poverty dataset. The amount of variance retained in the data increases as the number of components increases. The red dot in the graph indicates that for approximately 51 number of principal components the amount of variance retained by the data was 90 %. Thus, for the poverty dataset, the number of features were reduced from 143 to 51 with 90 % of the total variance still maintained.

6.2 t-distributed stochastic neighbor embedding (t-SNE)

Another technique for dealing with high dimensional dataset is called t-SNE. Similar to PCA, t-SNE is also a feature reduction technique that reduces the higher dimensional data to lower dimensional dataset. Unlike PCA which is a linear algorithm, t-SNE is a non-linear algorithm which can be used to find complex and non-linear dependencies in the dataset [35].

t-SNE is a probability based technique whereas, PCA is more mathematics based technique for feature reduction [12]. t-SNE internally computes a similarity matrix (similar to correlation matrix). t-SNE calculates a similarity score by calculating the distance between each point in the dataset to every other point in the dataset. Data points having larger similarity score indicates the dissimilarity between these data points. Therefore, dissimilar data points should not be clustered together. Whereas, data points having similarity distance closer to zero indicates that these data points are similar to each other and can be clustered together in the same cluster. Once the similarity scores for higher dimensional data is calculated, the data is projected onto the lower dimensional space usually 2D. Again, the similarity scores between the data points in lower dimensional space is calculated. Depending upon the similarity scores, the data points are grouped together. Usually, data points belonging to same target class are similar to each other and therefore, using t-SNE, such data points are clustered together [37].

t-SNE uses a parameter known as perplexity which controls the number of nearest neighbors that are to be considered while calculating the similarity matrix [37]. Larger datasets needs larger values for the perplexity parameter. Usually, perplexity lies in the range from 5 to 50[37]. Another parameter which t-SNE takes into consideration is the learning rate. Learning rate usually ranges from 10 to 1000[37]. When the learning rate is very less, the data points are cluttered together and when the learning rate is high, data points are well separated and are grouped together in the clusters of circular shape [37].

However, t-SNE is not recommended for very high dimensional dataset because after a certain point in time the learning becomes really slow and the amount of memory required also become very large [12]. Also, with the slow learning rate, probability of t-SNE of getting stuck in a local minima increases[12]. Hence, it is always recommended to use PCA before applying t-SNE on higher dimensional dataset. Thus, PCA would reduce the dimensions to some extent and then t-SNE would further decrease the number of dimensions.

Figure 32 shows the effect of applying t-SNE on the poverty data. It shows that all the poverty classes are not much separated into well defined clusters.

Figure 33 shows the effect of applying PCA before implementing t-SNE on the poverty data. It shows that classes are little separated into well defined clusters. This shows that applying PCA before t-SNE improve the results of t-SNE.

Figure 34 shows the effect of applying PCA before implementing t-SNE on the poverty data with a learning rate = 500.

Figure 35 shows the effect of applying PCA before implementing t-SNE on the poverty data with a perplexity = 20.

Figure 36 shows the effect of applying PCA before implementing t-SNE on the poverty data with a perplexity = 50.

Together the figures 32, 33, 34, 35, 36 shows that PCA and t-SNE does not works well when the dataset is categorical. Since, the poverty dataset is categorical, these techniques are not suitable for categorical type of datasets.

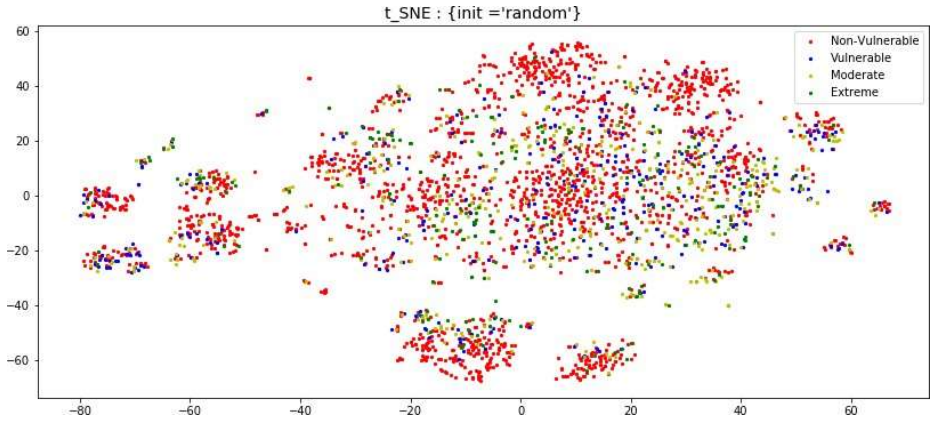

Fig. 32. Implementation of t-SNE on Poverty dataset [22]

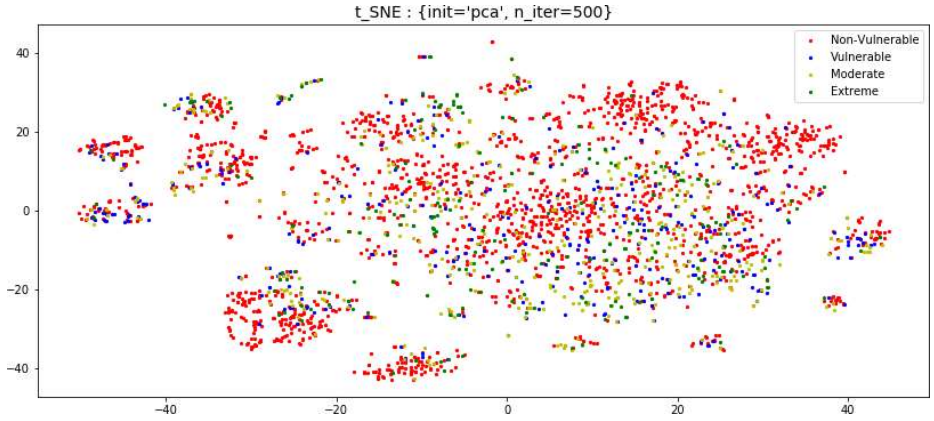

Fig. 33. Implementation of PCA and t-SNE on Poverty dataset[22]

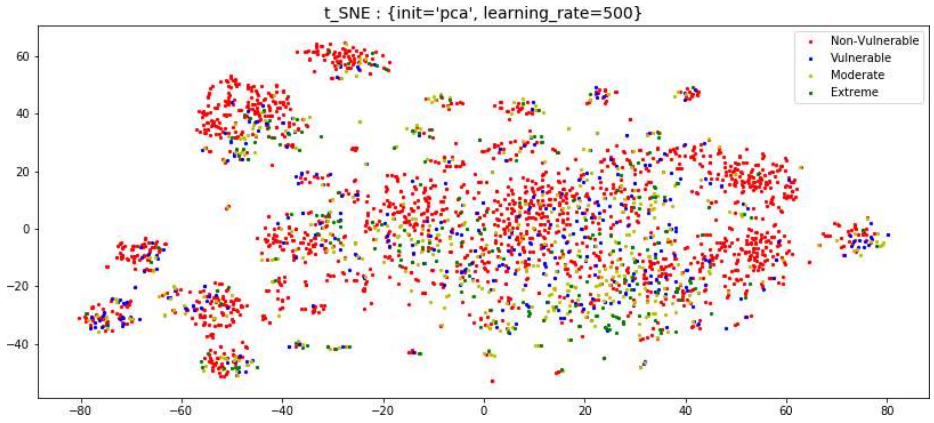

Fig. 34. Implementation of PCA and t-SNE on Poverty dataset with learning rate = 500 [22]

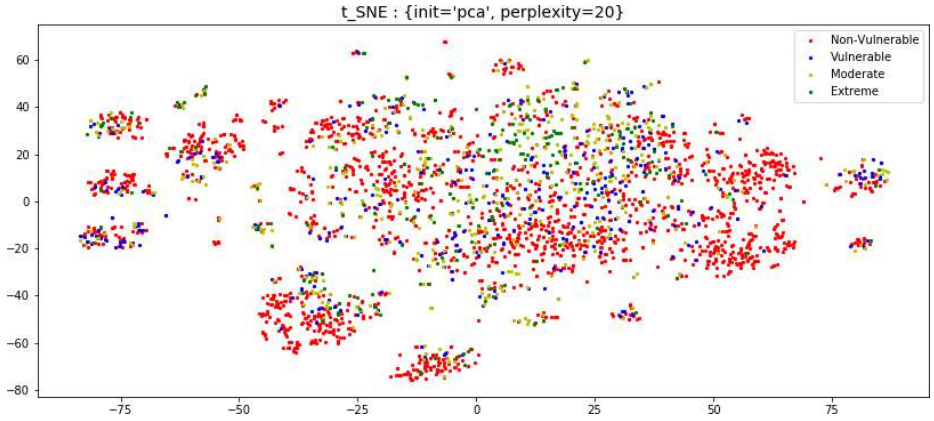

Fig. 35. Implementation of PCA and t-SNE on Poverty dataset with perplexity = 20 [22]

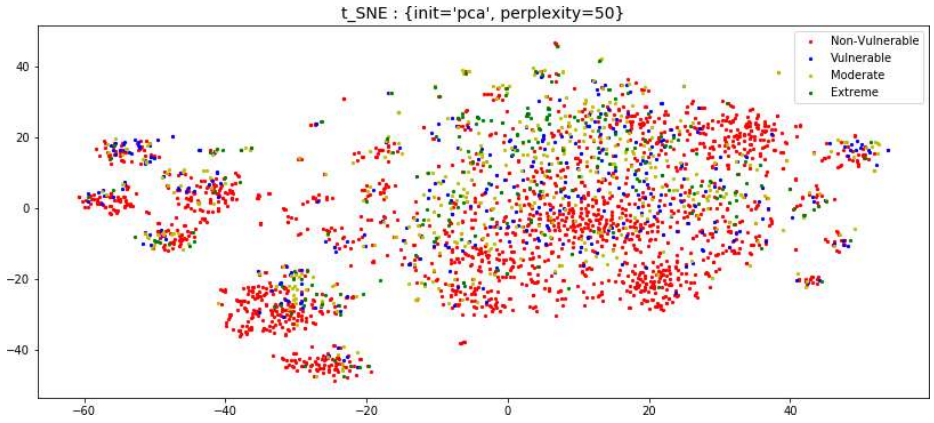

Fig. 36. Implementation of PCA and t-SNE on Poverty dataset with perplexity = 50[22]

Table 3. Time required for implementing t-SNE using several hyper-parameters

t-SNE parameters	Time in Sec
{init: random}	91.91
{init: pca, n iter: 500}	47.07
{init: pca, learning rate : 500}	87.95
{init: pca, perplexity : 20}	78.29
{init: pca, perplexity : 50}	121.51

Table 3 shows time (in seconds) required for implementing t-SNE for different set of hyper-parameters.

t-SNE works well for other datasets which are not categorical in nature, but since this dataset was categorical, t-SNE reveals very less information.

7 BALANCING THE TARGET CLASS

The imbalanced dataset is a very usual problem in Data Science. The dataset is imbalanced when there is an unequal number of observations for each of the target class in the dataset. When machine learning algorithms are applied to such an imbalanced dataset, algorithms might perform badly (resulting in lower accuracy or over-fitting) or may yield unpredictable or biased results.

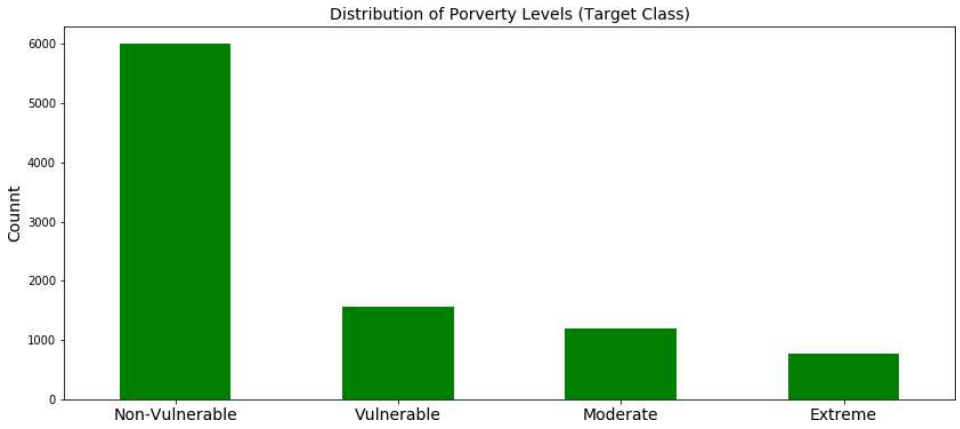

Fig. 37. Distribution of Target Variables - Imbalanced Dataset

Figure 37 shows imbalance in the distribution of target classes. To overcome this problem, the best solution is to have a balanced data. There are numerous techniques that can be implemented to achieve a balanced dataset. One of the technique is to gather more data. But, it is not always possible to collect more data to balance the dataset. Hence, this technique is not frequently used by the Data Scientists.

Other frequently used methods to overcome with class imbalance problem are random under-sampling and random over-sampling. Both these methods are discussed in sub-sections 7.1 and 7.2.

7.1 Under-Sampling the Dataset

The under-sampling technique works with the majority class in the dataset[7]. Minority class is a target class having a lesser number of observations when compared with the majority class (classes having more number of observations). Under-sampling techniques randomly discards the majority classes instances. This process continues till both majority and minority class observations become equal [7].

This technique is used when there is a big dataset and eliminating some of the observation does not have any impact on the performance of an algorithm[7]. The only disadvantage of using random under-sampling technique is that it may result in the loss of certain important information [7]. To perform random under-sampling, imblearn [25] package for python was used. In imblearn there is a method known as RandomUnderSampler() [25] which was used to randomly under-sample the majority class in the dataset.

Figure 37 indicates that non-vulnerable poverty level is a majority class in the dataset whereas vulnerable, moderate and extreme poverty levels are minority classes. RandomUnderSampler() [25]

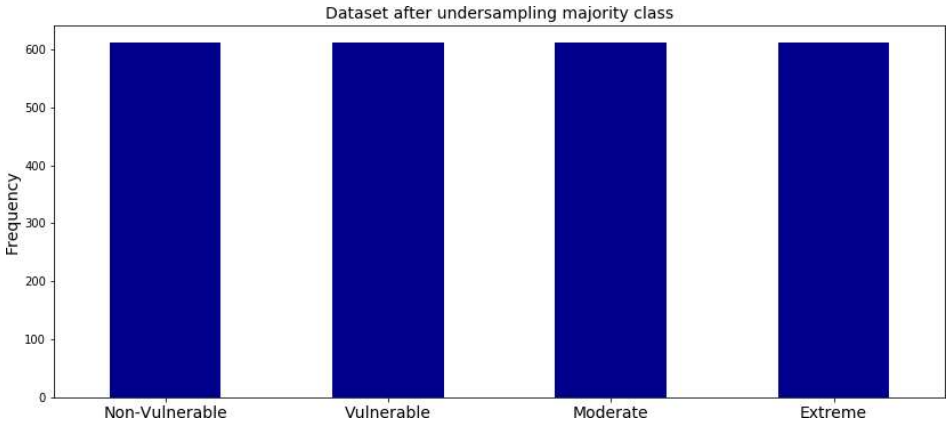

Fig. 38. Random under-sampling of the majority class values

tries to balance these minority classes with the majority class. Therefore, randomly few instances of non-vulnerable, moderate and vulnerable poverty levels were removed from the dataset until all the 4 target variables had the same number of total observations.

Figure 38 shows the distribution of target class after applying random under-sampler. Since the target class "extreme" had the lowest number of observations i.e. around 600 all the other class was reduced to around 600 observations. Therefore, after performing the random under-sampling technique, all the 4 target poverty levels had the same number of total observations i.e., around 600.

7.2 Over-Sampling the Dataset

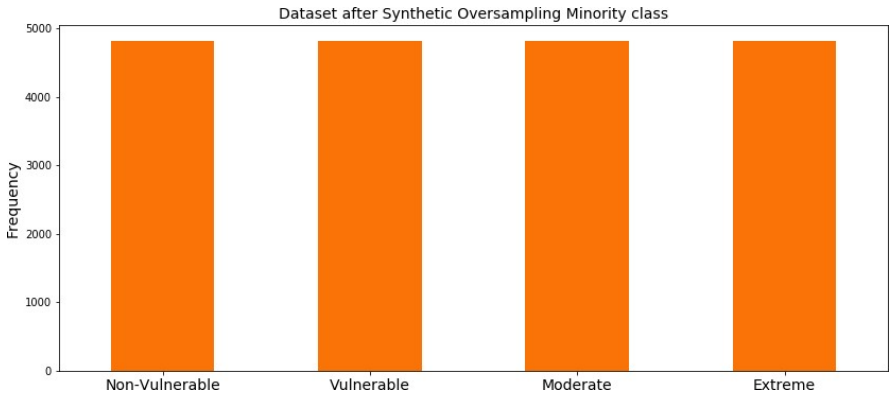

Fig. 39. Over-Sampling the minority Class using SMOTE technique

The over-sampling technique works with the minority class in the dataset. One of the most popular over-sampling technique used is known as Synthetic Minority Over-sampling (SMOTE).

SMOTE analysis randomly generates synthetic data that is almost related to the subset of the data belonging to the minority class [7]. This process continues till both majority and minority class observations become equal. Therefore, instead of removing instances that belong to the majority class as described in section 7.1, SMOTE over-sampling technique adds new synthetic data into the dataset.

The advantages of using this technique are that it does not lead to loss of important information from the dataset, unlike random under-sampling technique. The only disadvantage of using SMOTE is that it may overfit the data as new synthetic data is being added for balancing the target class 7.1. SMOTE() [24] method from imblearn [24] was used to over-sample the minority class in the dataset.

Figure 37 indicates that non-vulnerable poverty level is a majority class in the dataset whereas vulnerable, moderate and extreme poverty levels are minority classes. SMOTE()[24] tries to balance these minority classes with the majority class in the dataset. Therefore, synthetic data almost similar to the subset of non-vulnerable, moderate and vulnerable class observations were created until all the 4 target variables had the same number of total observations.

Figure 39 shows the distribution of target class after applying SMOTE()[24]. Therefore, after applying SMOTE(), the total number of observations for all the minority classes were increased and all the 4 target poverty levels had the same number of total observations around 5000.

8 SPLIT DATASET IN TRAIN AND TEST

The entire dataset was divided into two parts. 80% of the total data was allocated for training the machine learning model (so that built model could be able to generalize the future unseen data) whereas, the remaining 20 % of the data was utilized for testing and making predictions using the models built.

train_test split() [29] function from Scikit-learn package was used to split the data into train and test datasets.

$$X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(features, Target, test\_size = 0.20, shuffle = True) \quad (8)$$

From the equation 8 [29] 'X train 'and 'X test 'denotes the feature set for training and testing the model whereas, 'Y_ train 'and 'Y_ test'denotes the set of target values for training and testing the model. 'test_size = 0.20 'signifies that 20% of the entire data is reserved for testing the built model and 'shuffie = True 'means that the dataset is randomly shuffled every time training and testing data are selected.

9 DATA NORMALIZATION AND STANDARDIZATION

Data normalization is extremely essential process in Data Science. Different features in the dataset may have different scales. For example, some attributes have range[0,1] while some other attributes may have a range [100, 1000]. When such type of dataset is used for training a machine learning algorithm especially K-Nearest Neighbor (KNN), the algorithm may produce biased results[4]. The Euclidean distances computed by K-Nearest Neighbor algorithm would be shifted closer to the features having larger scale [4]. Hence, when KNN is trained with such non-uniformly scaled dataset, produced biased and erratic results. Therefore, it is important to rescale the dataset[4].

Data normalization techniques rescales the entire dataset in such a way that all the features in the dataset have same range of scale[4]. Thus, data normalization helps to achieve a common basis of comparison. Usually, features are scaled in such a way that, all the features in the dataset have a mean $\mu = 0$ and variance $\sigma = 1$[4]. There are several methods that can be utilized for normalizing the data. Some of them includes Min-Max Scaler and Z-score or Standard Scaler.

Min-Max Scaler normalizes the data in the range [0,1] or [-1, 1]. It uses the equation 9 [4].

$$z = \frac{x - min(x)}{[max(x) - min(x)]} \quad (9)$$

Z-score or Standard Scaler rescales the data such that mean $\mu = 0$ and variance $\sigma = 1$[4]. Thus, after rescaling the dataset, the data is normally distributed having a bell-shaped curve. It uses the equation 10 [4].

$$z = \frac{x_i - \mu}{\sigma} \quad (10)$$

For this independent study, Z-score or Standard Scaler was used to rescale the dataset. For this purpose, Pipeline [29] from Sklearn was implemented to apply standard scalar to the entire dataset. Pipeline sequentially applies a series of functions (in the order they are written) to the entire dataset.

$$Pipeline([(imputer, Imputer(strategy = median)), (scaler, StandardScaler([0, 1]))]) \quad (11)$$

Equation 11 [29] indicates that, the pipeline first imputes the missing data (in any) as the median value and then applies Standard Scaler to the dataset to rescale the data to the default range [0,1] and finally returns the transformed dataset.

10 PARAMETER SELECTION WITH GRID SEARCH

Grid Search is a technique that helps to tune different hyper-parameters of machine learning in order to achieve optimal results without over fitting the data. GridSearchCV() [29] from scikit learn was used to perform hyper-parameter tuning.

In a grid search, a grid is defined consisting of various parameters for the given algorithm. A k-fold cross-validation technique is used internally to perform an extensive search on the grid and to decide on the most optimal hyper-parameters for that given algorithm[28]. The grid is nothing but a dictionary of hyper-parameters that are to be tuned while training the model. Keys in the dictionary denote several parameters and values of those keys indicates several possible values for that parameters [28].

Code 1 shows a sample grid for Random Forest algorithm. For example "max_depth" is a parameter for Random Forest which indicates maximum depth of the tree and [5, 6, 7, 9] shows all possible values for maximum depth of the tree.

Listing 1. Example of the grid structure for GridSearchCV algorithm [29]

```
param_grid = {'n_estimators': [30, 35, 50, 80],
              'max_depth': [5, 6, 7, 9],
              'min_samples_leaf': [1, 5, 10, 15],
              'max_leaf_nodes': [5, 10, 20, 30]}
```

Equation 12 [29] is a GridSearchCV equation from scikit learn. In this equation, "***estimator = RandomForest()***" indicates to implement and tune the hyper-parameters of a Random Forest algorithm. "***param_grid = param_grid***" shows the grid of parameters (set in the Code 1) that are to be hyper tuned. "***cv = 5***" indicates that 5-fold cross validation will be performed while searching for the most optimal hyper paramters. "***n_jobs = -1***" indicates to use all the cores of the CPU for faster performance.

$$GridSearchCV(estimator = RandomForest(), param\_grid = param\_grid, cv = 5, n\_jobs = -1) \tag{12}$$

The GridSearchCV() returns best_params_ and best_scores_ [28]. best_params_ indicates optimal set of hyper-parameters and best_scores gives the best cross-validation score.

Listing 2. Example of the most optimal hyper-parameters returned by Grid Search for Random Forest algorithm[29]

```
RandomForestClassifier(n_estimators=50, criterion='gini', max_depth=9,
    min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features
    ='auto', max_leaf_nodes=30, min_impurity_decrease=0.0, min_impurity_split=None,
    bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0,
    warm_start=False, class_weight=None)
```

Code 2 shows the optimal set of hyper parameters returned by the grid search algorithm.

11 MACHINE LEARNING MODELS

Various supervised machine learning algorithms were implemented for classifying the families to one of the poverty levels such as non-vulnerable, vulnerable, moderate, and extreme.

11.1 Naive Bayes

Naive Bayes uses different types of probabilities for classifying the data into different classes. In Naive Bayes, every feature/attribute in the dataset contributes individually for classifying all the instances in the data. Hence, it assumes that the features in the dataset are statistically independent from each other[34]. Naive Bayes calculates posterior probability given accesses to some prior knowledge[34]. That is, it calculates the probability of a target class given a set of features (which are statistically independent from each other) [34].

The algorithm uses the equation 13 [34] for calculating the posterior probability. In the equation 13 P(x—c) = probability of features given a target class, P(c) = prior probability and P(x) = normalizing constant [34].

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{13}$$

For this independent study, Naive Bayes was implemented on 3 types of datasets (raw, under-sampled and over-sampled). The dataset was split into training and testing with ratio 80:20. Grid Search was implemented to find the most optimal hyper-parameters. Since, poverty prediction was a multi-class problem BernoulliNB() [29] (suitable for multi-class problems) from scikit learn was used to implement Naive Bayes

Listing 3. Best hyper-parameters for Naive Bayes using GridSearchCV() [29]

```
BernoulliNB (alpha = 1, binarize = True, fit_prior = True, class_prior = None)
```

Code 3 shows the most optimal set of parameters returned by GridSearchCV(). It shows that the algorithm used alpha = 1 to avoid zero probability problem while implementing Naive Bayes. Since, the dataset was almost binary, binarize parameter was set to True.

The performance of Naive Bayes algorithm on different types of datasets (raw, under-sampled and oversampled) is discussed in the section 13.

11.2 Random Forest

Random Forest is an ensemble classification algorithm. The main aim of ensembles is to produce better results than non-ensembles without over-fitting the data. Random Forest combines several weaker classifier such as decision trees to generate a strong classifier[14]. Thus, it combines several decisions trees (weak classifier) to create a forest [14]. Several decision trees are built and each of the decision tree in the forest tries to classify the unseen data instance. At the end, a majority voting algorithm is used. The output that receives majority of votes is used as the final target class for that particular unseen instance [14].

For this independent study, Random forest was also implemented on 3 types of datasets (raw, under-sampled and over-sampled). The dataset was split into training and testing with ratio 80:20. Grid Search was implemented to find the most optimal hyper-parameters.

Listing 4. Best hyper-parameters for Random Forest using GridSearchCV()[29]

```
RandomForestClassifier(n_estimators=50, criterion='gini', max_depth=9,
    min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features
    ='auto', max_leaf_nodes=30, min_impurity_decrease=0.0, min_impurity_split=None,
    bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0,
    warm_start=False, class_weight=None)
```

Code 4 [14] shows the most optimal hyper-parameters that gives the best accuracy for Random Forest algorithm. It showed that the algorithm allowed maximum of 30 leaf nodes in a given forest. Each decision tree built was restricted to have a maximum depth of 9. Gini was by the decision tree to measure the impurity of a given node and maximum number of decision trees allowed in the forest was 50.

The performance of Random Forest algorithm on different types of datasets is also discussed in the section 13.

11.3 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a supervised machine learning algorithm which is used for classifying the data. KNN uses the target class values of its K most nearest neighbors for classifying the unseen instances from testing or validation set[9]. A majority voting algorithm is executed on the K most nearest neighbors of a testing instance, and the most common target value among its k nearest neighbors is assigned to the given unseen instance [9].

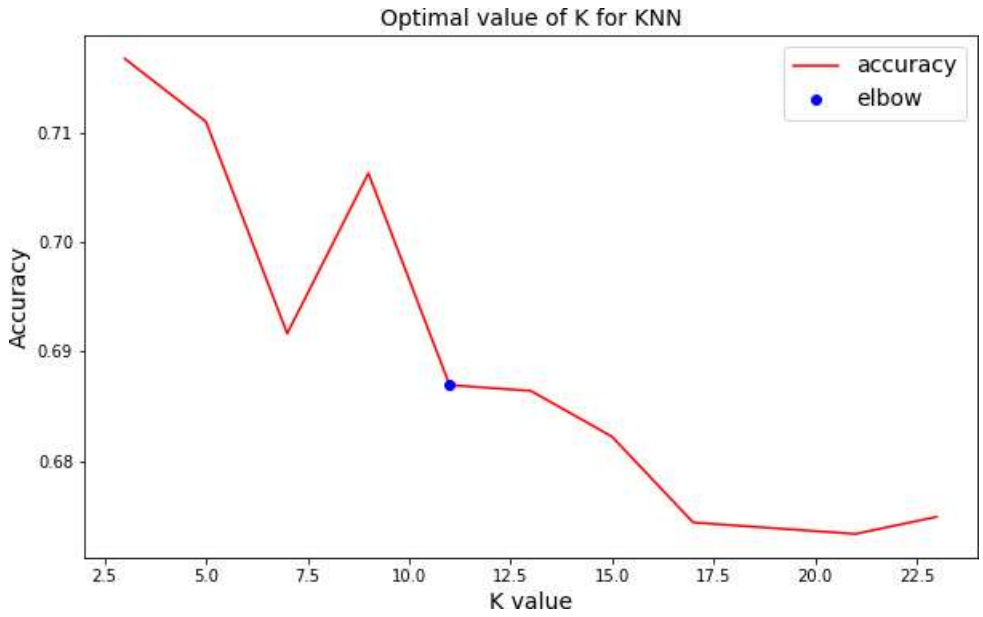

Fig. 40. Elbow Method for finding most optimal value of K for KNN algorithm

It is very important to choose the correct value of K for KNN. It may cause over fitting if the value of K chosen is very small[36]. Whereas, there are chances of mis-classifying the observations

if K is chosen to have a very larger value[36]. Usually, to avoid ties and to avoid unpredictable results during the classification, K is always chosen to have a perfect odd value. An elbow/knee method is implemented to select the best and the most optimal value of K for training the KNN model [36].

Figure 40 shows optimal value of K was selected as 11 for classifying families into different poverty levels. The accuracy scores for different values of K was plotted and the elbow or knee point was chosen as the most optimal value of K. Elbow/Knee point is the point on the graph 40 after which the accuracy either decreases continuously or remains constant[36].

For this independent study, KNN was implemented on 3 types of datasets (raw, under-sampled and over-sampled). The dataset was split into training and testing with ratio 80:20. Grid Search was implemented to find the most optimal hyper-parameters.

Listing 5. Best hyper-parameters for KNN using GridSearchCV()[29]

```
KNeighborsClassifier (n_neighbors=11, weights='uniform', algorithm='ball tree',
    leaf_size=25, p=2, metric='minkowski', metric_params=None, n_jobs=-1)
```

Code 5 shows the most optimal hyper-parameters for KNN algorithm. It shows that the algorithm used 11 k-nearest neighbors and minkowski distance with p = 2 for classification.

The performance of KNN algorithm on different types of datasets is discussed in section 13.

11.4 Support Vector Machines (SVM)

SVM was another supervised machine learning algorithm implemented for predicting the poverty levels. SVM separates the observations belonging to different classes by fitting a hyper-plane [16]. In two dimensional space, a straight line is used for classifying unseen instances. All the instances on one side of hyper-plane belongs to one set of target class while all the instances on the other side of hyper-plane belongs to other set of target class. The most optimal hyper-plane is the one that maximizes the distances between the two set of target classes i.e. maximizes the margin [16].

Not necessary, every-time the dataset is linearly separable[32]. Therefore, a straight line would not be always efficient to separate the observations that belong to different class. To well-separate the non-linear data, SVM uses kernel tricks[32]. Thus, kernels helps SVM to convert the non-separable data into separable set of data points [32].

For this independent study, Support Vector Machine (SVM) was also implemented on 3 types of datasets (raw, under-sampled and over-sampled). The dataset was split into training and testing with ratio 80:20. Grid Search was implemented to find the most optimal hyper-parameters. SVM gave the best accuracy as f1-score (for a given type of dataset) when implemented with the most optimal set of hyper-parameters. LinearSVC [29] supports multi-class implementations. Since, poverty prediction was a multi-class problem LinearSVC was implemented for this independent study.

Listing 6. Best hyper-parameters for SVM using GridSearchCV()[29]

```
LinearSVC(penalty='l2', loss='hinge', dual=True, tol=0.0001, C=2.0, multi_class='ovr',
    fit_intercept=True, intercept_scaling=3, class_weight=None, verbose=0,
    random_state=True, max_iter=500)
```

Code 6 shows the most optimal hyper-parameters that gives the best accuracy for SVM algorithm. It indicates that the algorithm used l2 penalization as a regularization method to avoid over-fitting on the training data. To penalize incorrectly classified instances C was set to 2. Since, poverty

prediction was a multi-class problem, multi class was set to_"ovr" and 500 were used as maximum number of iterations for executing linear SVM.

The performance of Support Vector Machine (SVM) algorithm on different types of datasets is also discussed in the section 13.

12 DEEP LEARNING MODELS

Keras [3] deep learning package for Python was used for implementing the neural network for predicting the poverty. A simple dense neural network with one input and one output layers together with one hidden layer was built for predicting the poverty. 70% of the entire data was utilized as train data and 30 % of the data was utilized as test / validation data for deep learning model.

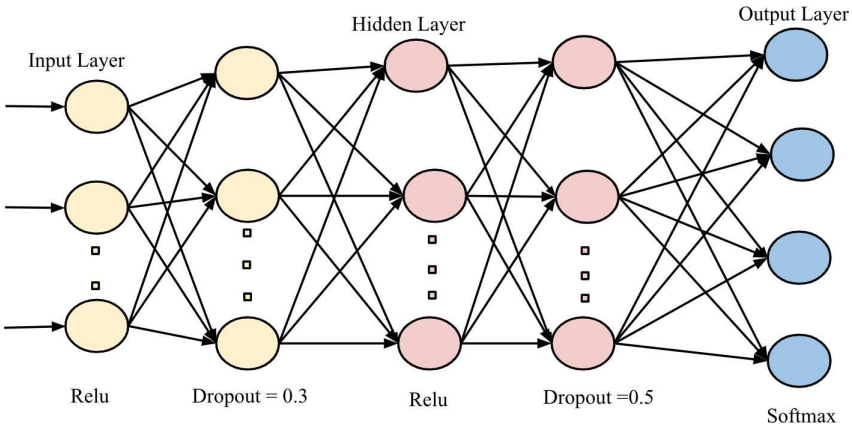

Fig. 41. Architecture of Deep Learning model for Predicting Poverty

Figure 41 depicts the architecture of deep learning model for predicting poverty.

Code 7 shows the overall structure and the list of activation functions used for building a deep learning model.

Listing 7. Structure of Deep Neural Network [3]

```
model = Sequential ()

model.add(Dense(64, input dim=input, kernel initializer ='uniform' , activation =' relu ' ) )
model.add(Dropout(0.30) )
model.add(BatchNormalization(momentum=0.99, epsilon=0.001,
 moving mean initializer =' zeros ' , moving variance initializer ='ones' ) )

model.add(Dense(64 , kernel initializer ='uniform' , activation =' relu ' ) )
model.add(BatchNormalization( moving mean initializer =' zeros ' , epsilon =0.001,
momentum=0.99, moving variance initializer ='ones' ) )
model.add(Dropout(0.50) )
```

```
model.add(Dense(output, kernel_initializer ='uniform', activation ='softmax' ) )
```

From code 7, the deep neural network was built by simply arranging all the layers one after the another [10]. For this purpose Sequential() function from Keras package was utilized. Different layers are added to the deep neural network simply by calling add() function. While building the input layer it is important to define the input_dim parameter. input_dim defines the number of inputs for the input layer [10]. Since there were 143 poverty features in the dataset, input_dim was set to 143. Similarly, it is also important to define the output parameter for the output layer. Usually, output parameter is assigned a number equal to the number of target classes. Since there were 4 target classes for poverty prediction, output was set to 4 [10].

Whether the layer of neural network is fully connected or whether it is conventional or whether it is bi-directional, it is defined in the add() function written in the code 7. A Dense class [10] in the add() function indicates that a fully connected layer is added into the neural network. Code 7 shows that the Dense class takes different number of arguments. The first parameter is the the number of neurons in each layer of the neural network (indicated by 64), the second parameter is the kernel_initializer = 'uniform' which uniformly initializes the weights for the neural network and a activation function. Input layers and all the hidden layers used relu as the activation function (achieves faster learning than tanh activation functions). Since, poverty prediction was a multi-class problem softmax activation was used in the final output layer [10].

In between every dense layer in the neural network, a dropout layer with probability of (0.30 and 0.50) was added. Dropout layers helped to reduce the over-fitting while training a deep neural network. Batch Normalization in the code 7 helped to rescale the input features in the scale form 0 to 1 so that the performance of the deep learning model is boosted.

Listing 8. Compile Deep Learning Model and set the Learning Rate [3]

```
model.compile(loss='mean squared error' , optimizer = Adam(lr=0.00005, decay=0.00001) ,
    metrics=[ 'accuracy' ])
```

Code 8 shows the settings needed for compiling a deep learning model. There are a set of parameters which are required for compiling the deep learning model. One of the most important parameter was the Optimizer. The optimizers update the weights of the deep learning model so as to reduce the over-fitting. Adaptive Moment Estimation (Adam) [20] which is a gradient descent algorithm was used as an optimizer with a learning rate of 0.0005 and decay of 0.00001. Decay indicated how the learning rate was reduced each time the weights were changed. Along with this, accuracy was used as the performance metric and mean_squared_error was used to measure the losses at the time of training [3].

Listing 9. FiNing a Deep Learning Model for training[3]

```
model. fit ( X_train , Y_train , epochs=30, batch size =64)
```

Code 9 shows that fit() function is used to train a deep learning model. The deep learning model would be trained for 30 iterations (denoted by epochs) of training data with a batch_size = 64. batch_size indicates the number of observations that are evaluated from the training data before the weights of the deep learning model are updated [10].

Table 4 shows how the deep learning model was trained for 30 epochs. It also shows the performance of the deep learning model in terms of accuracy and loss encountered for each of the 30 epochs. For example, in epoch = 1, the model had a loss of 0.44 and an accuracy of 80%. The accuracy of the model was increased gradually with the increase in the number of epochs. For epoch = 7 the accuracy of the deep learning model was around 85% with loss of 0.36.

Table 4. Output performance on each epochs for Deep Learning Model[3]

Epcohs	Loss	Accuracy
1	0.5	0.83
2	0.47	0.83
3	0.44	0.88
4	0.40	0.88
5	0.37	0.90
6	0.36	0.90

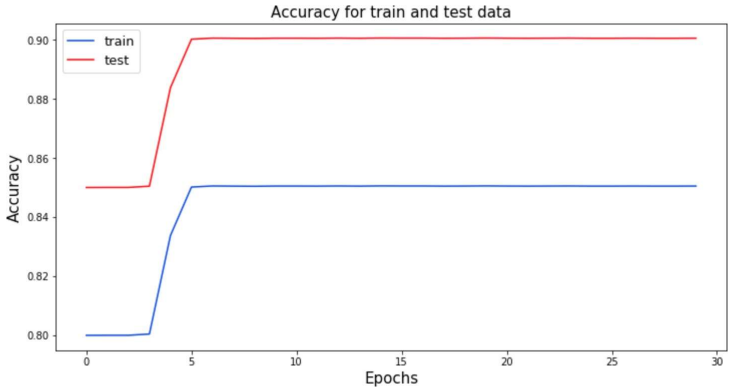

Fig. 42. Accuracy on train and test data for Deep Learning model

Figure 42 shows the performance of the deep learning model on training and testing datasets for 30 epochs. The model could achieve a training accuracy of approximately 85 % and a testing accuracy of around 89%. The performance of the model can be increased by training the neural network with more data or by adding more convolution hidden layers into the neural network.

Table 5 shows in detail the list of hyper-parameters used for building the deep learning model.

Table 5. List of Hyper-Parameters for Deep Neural Network

Paramater	Values
Batch size	64
Epochs	30
Learning Rate	0.00005
Decay	0.00001
Optimizer	Adam
Momentum	0.99
Activation Function	Relu/Softmax
Regularization	Dropout (0.30, 0.50)
Loss Function	mean squared error
Metrics	Accuracy

13 EVALUATING RESULTS

This section describes in detail the performance of several machine learning and deep learning models. Several machine learning and deep learning models are compared based on both accuracy and f1-scores. Usually, the f1-score is considered a better evaluation metric than the accuracy score because f1-score takes into consideration both precision and recall while evaluating the models. The performances of different models when implemented using the raw dataset, under-sampled dataset, and over-sampled dataset is also compared.

13.1 Performance based on Raw Dataset

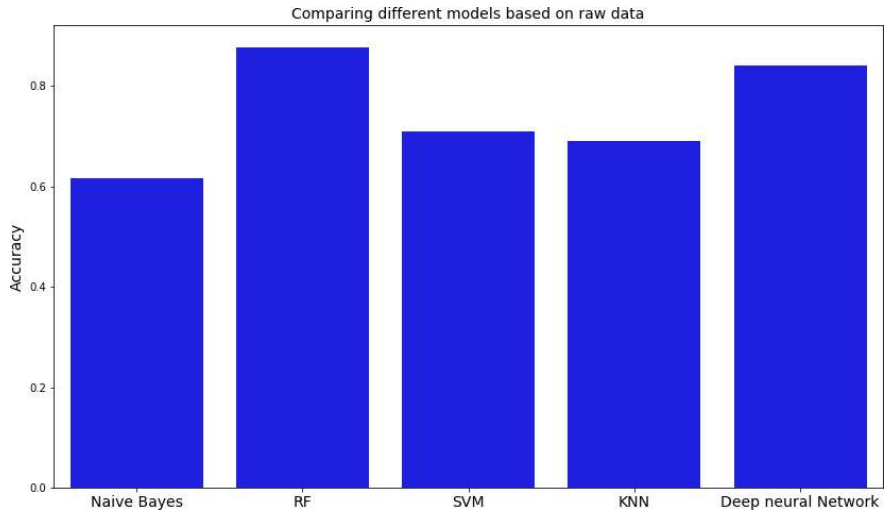

Fig. 43. Accuracy of models when implemented using the Raw Dataset

Figure 43 shows the performances (in terms of accuracy) of Naive Bayes, Random Forest, Support Vector Machine, K-Nearest Neighbor, and Deep Neural Network when trained using raw dataset. The raw dataset is not a balanced dataset i.e. number of observations for all the target class is not same. This imbalance in the dataset has caused algorithms like Naive Bayes, SVM, and KNN to perform moderately well.

Figure 43 indicates that Random Forest which is an ensemble of Decision tree outperforms all the remaining models with an accuracy of 87%. Deep Neural Network also performed well with an accuracy of 85%. Among all the algorithms implemented, Naive Bayes performed worst on the raw dataset and achieved an accuracy of 64%.

Figure 44 shows the performances (in terms of f1-score) of Naive Bayes, Random Forest, Support Vector Machine, and K-Nearest Neighbor when trained using raw dataset. Figure 44 indicates that Random Forest again outperforms all the remaining models with an f1-score of 0. 76. SVM and KNN achieved an f1-score of 0.70 and 0.69 respectively. Again among all the algorithms implemented, Naive Bayes performed worst on the raw dataset and achieved an f1-score of 0.65.

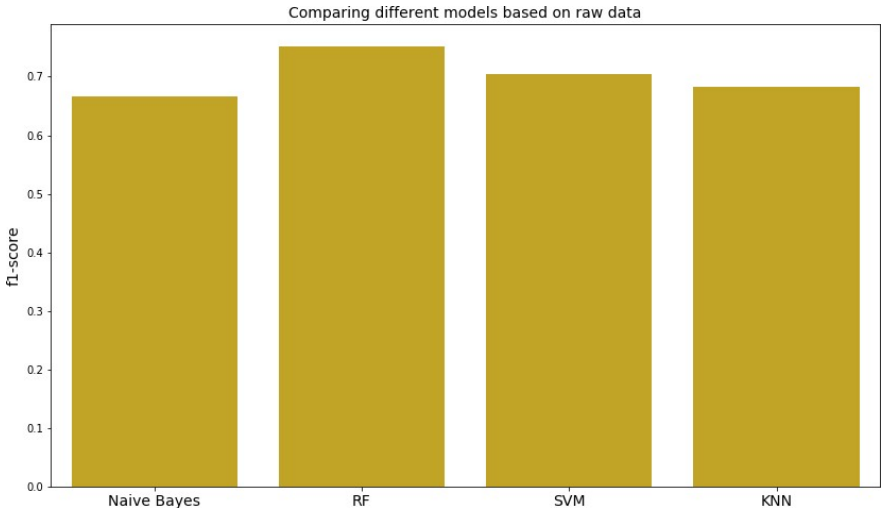

Fig. 44. F1-Score of models when implemented using the Raw Dataset

13.2 Performance based on Under-Sampled Dataset

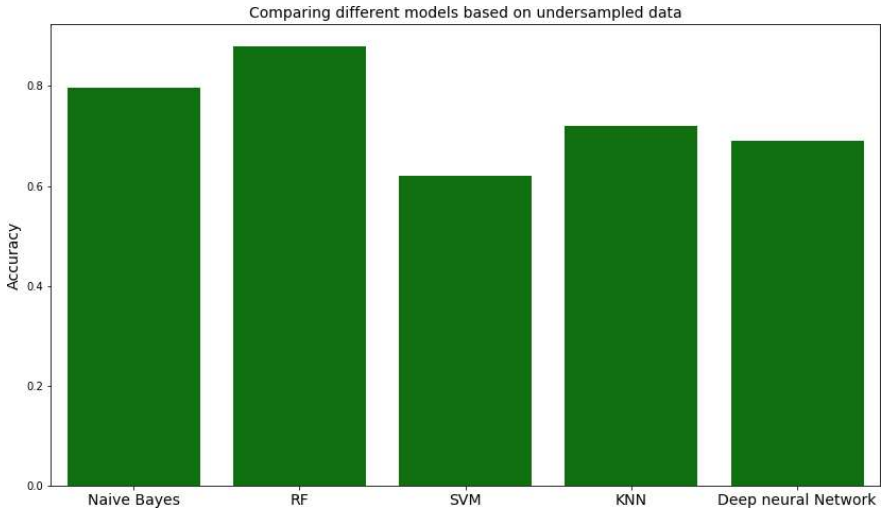

Fig. 45. Accuracy of models when implemented using the Under Sampled Dataset

Figure 45 shows the performances (in terms of accuracy) of Naive Bayes, Random Forest, Support Vector Machine, K-Nearest Neighbor, and Deep Neural Network when trained using under-sampled dataset. In under-sampling, observations from majority target class are randomly removed in such a way that number of observations for each target class becomes equal. Any deep learning model requires larger amount of data for training and since, under-sampling reduced the dataset, the accuracy of deep learning model was reduced to 69 % when compared with the raw dataset.

Figure 45 indicates that Random Forest which again outperforms all the remaining models with an accuracy of 88%. Naive Bayes and KNN performed decently and achieved an accuracy of 79% and 72 % respectively. Among all the algorithms implemented, SVM performed worst on the under-sampled data and achieved an accuracy of 62%.

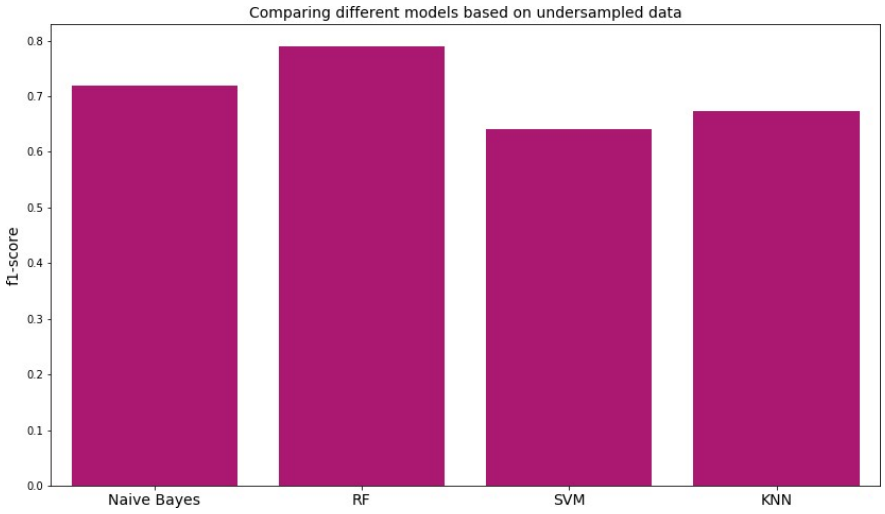

Fig. 46. F1-Score of models when implemented using the Under Sampled Dataset

Figure 46 shows the performances (in terms of f1-score) of Naive Bayes, Random Forest, Support Vector Machine, and K-Nearest Neighbor when trained using under-sampled dataset. Figure 46 indicates that Random Forest again outperforms all the remaining models with an f1-score of 0. 79. Naive Bayes and KNN achieved an f1-score of 0.72 and 0.67 respectively. Again among all the algorithms implemented, SVM performed worst on under-sampled data and achieved an f1-score of 0.64.

13.3 Performance based on Over-Sampled (SMOTE) Dataset

Figure 47 shows the performances (in terms of accuracy) of Naive Bayes, Random Forest, Support Vector Machine, K-Nearest Neighbor, and Deep Neural Network when trained using over-sampled (SMOTE) dataset. In over-sampling, synthetic data similar to minority class is randomly added into the dataset such that number of observations for each target class becomes equal. Any deep learning model requires a larger amount of data for training and since, over-sampling increased the total size of the dataset, the accuracy of deep learning model was increased to 90 % when compared with the performance of deep learning model on raw and under-sampled dataset.

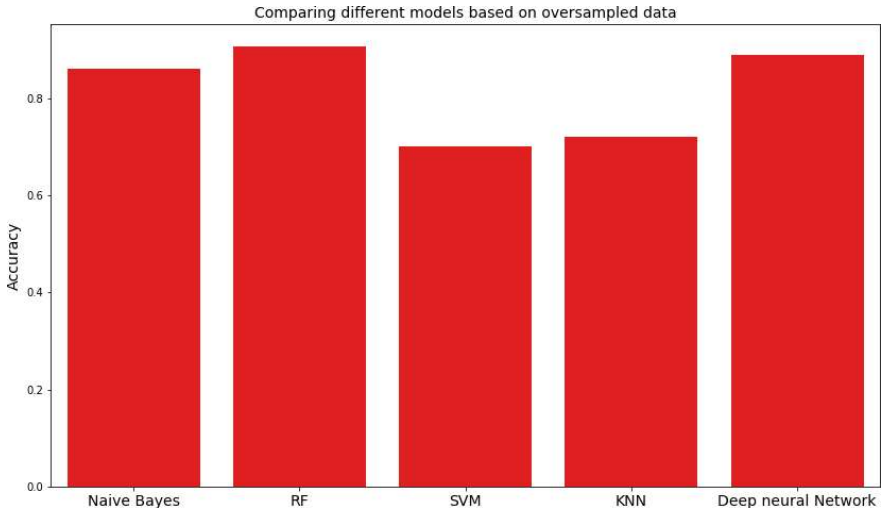

Fig. 47. Accuracy of models when implemented using the Over Sampled (SMOTE) Dataset

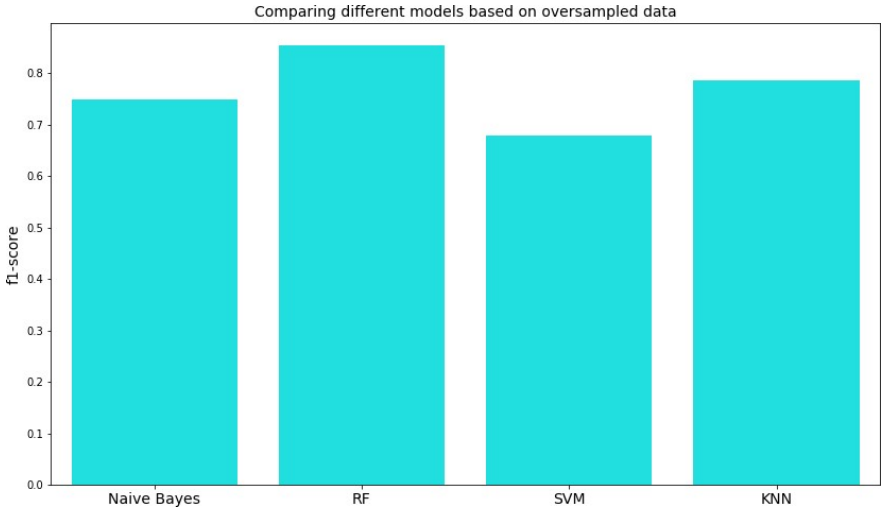

Fig. 48. F1-Score of models when implemented using the Over Sampled (SMOTE) Dataset

Figure 47 indicates that the Random Forest which again outperforms all the remaining models with an accuracy of 90%. The performance of Naive Bayes was also improved to an accuracy if 86% compared to its performance on the raw data and under-sampled dataset. KNN and SVM

performed decently and achieved an accuracy of 72% and 70% respectively. Among all the algorithms implemented, SVM performed worst on the under-sampled data and achieved an accuracy of 62%. Figure 48 shows the performances (in terms of f1-score) of Naive Bayes, Random Forest, Support Vector Machine, and K-Nearest Neighbor when trained using the over-sampled dataset. Figure 48 indicates that Random Forest again outperforms all the remaining models with an f1-score of 0. 85. Naive Bayes and KNN achieved an f1-score of 0.75 and 0.78 respectively. Again among all the algorithms implemented, SVM performed worst on under-sampled data and achieved an f1-score of 0.68.

13.4 Overall Results

Table 6 compares the performances of different models in terms of accuracy.

Table 6. Performance (accuracy) of different models based on different versions of dataset

Model	Raw Dataset	Under-Sampled	Over-Sampled
Naive Bayes	0.617	0.796	0.86
Random Forest	0.87	0.88	0.91
SVM	0.71	0.62	0.70
KNN	0.69	0.72	0.72
Neural Net.	0.84	0.69	0.89

Table 7 compares the performances of different models in terms of f1-score.

Table 7. Performance (f1-scores) of different models based on different versions of dataset

Model	Raw Dataset	Under-Sampled	Over-Sampled
Naive Bayes	0.67	0.72	0.75
Random Forest	0.75	0.79	0.86
SVM	0.70	0.64	0.68
KNN	0.68	0.67	0.79

14 SENTIMENTAL ANALYSIS OF POVERTY RELATED TWEETS

14.1 Collecting Poverty related Tweets

Tweets related to poverty were scraped from Twitter. Tweepy Search API [33] for Python was utilized for extracting around 15000 poverty related tweets. Different #hashtags and @usernames related to poverty were used as the keywords by Tweepy [33] for extracting tweets. Different keywords used for extracting poverty related tweets are mentioned in the table 8

Table 8. List of #hashtags and @usernames used for extracting poverty related tweets

Keywords
#poverty
#poor
#extremepoverty
#endpoverty
#povertylevel
#income
#lowincome
#noeducation
#homeless
#charity
#hunger
#zerohunger
@WorldBank

14.2 Preprocessing Poverty related Tweets

All the tweets extracted were pre-processed and cleaned for sentimental analysis task. Different data cleaning operations performed is listed below [18].

- All the extracted tweets were converted to standard ascii and utf-8 encoding format[18].
- All the white spaces were removed (both leading and trailing).
- All the occurences of @usernames, # symbols and numbers were removed[18].
- All the URL links and re-tweet symbols (RT) in the tweets were removed[18].
- All punctuations symbols were eliminated[18].
- Extra white spaces within two words was eliminated and was replaced with just a single space[18].

Apart from cleaning the tweets different Natural Language Processing techniques were also implemented. Natural Language ToolKit (NLTK) [5] was used for performing several NLP tasks. Following is the list of several NLP tasks.

- All the words in the tweets were converted to lower case letters.
- All the stop words [5] from the tweets were removed.
- All the tweets were tokenized [5] into individual tokens.
- All the words were converted to their root form using Lemmatization [5]. For example, words like playing, plays, played were converted to their root form play.
- All the incorrect spelling [5] of different words in the tweets were corrected

14.3 Sentimental Analysis

Sentimental Analysis is useful for understanding emotions and sentiments of several thousand people. Sentiments are categorized into 3 polarities called positive, negative, and neutral. Sentimental Analysis with poverty data can help us to understand the sentiments of people belonging to the families that are prone to poverty and what are the measures that are being used to eliminate world poverty.

Since the dataset of different poverty related tweets was not labeled, an unsupervised sentimental analysis technique was implemented for identifying positive, neutral and negative sentiments. TexBlob an API for pyhton [26] was used to perform sentimental analysis on poverty tweeets. TextBlob [26] is a dictionary and rule based sentimental analysis tool that uses a dictionary of words for classifying tweets as positve, negative or neutral.

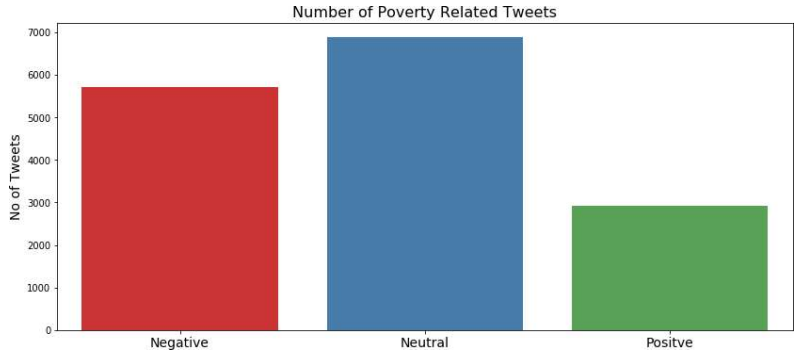

Fig. 49. Total number of Negative, Neutral and Positive Poverty Tweets a er Sentiment Analysis

After performing sentimental analysis, TexBlob API returns 2 values knowns as polarity and subjectivity[26]. Polarity takes value in range [-1, +1] where -1 shows extreme negativity and +1 shows extreme positivity. A polarity of 0 indicates a neutral sentiment. Subjectivity takes value in range [0,1] where 0 indicate that the tweet is more objective lacking emotions and sentiments whereas subjectivity = 1 indicates that the tweet is full of emotions and sentiments[26].

Figure 49 shows the total number of Negative, Neutral and Positive Tweets after performing Sentiment Analysis on Poverty data. Majority of the poverty tweets were classified either as having negative sentiments or neutral sentiments. Around 3000 tweets were classified as positive using TexBlob sentimental analysis tool.

14.4 Word Cloud for Positive and Negative Poverty Tweets

Word Clouds are a visualization tool that is extremely important when one wants to visualize the textual data. Word Clouds examines the text data to detect trends. Word cloud determines the frequency of occurrence of every word in the text data[15]. All the frequently appearing words are given bigger font sizes and are highlighted[15]. Whereas, all the less frequently appearing words are given smaller font sizes and are not highlighted[15]. Therefore, word cloud helps to interpret trends and patterns in the text data. With word cloud, it is easy to keep track of all the frequently occurring words. This would have been hard with tables or graphs containing a list of frequently occurring words[15].

Text data has to be preprocessed before it can be used by a word cloud[15]. Word clouds do not understand the importance of different words in the text data, therefore, they are best suited for performing an initial analysis of data[15].

A wordcloud package for Python was used to create word clouds for both positive and negative tweets related to poverty.

Fig. 50. WordCloud for negatively classified poverty tweets

Figure 50 shows a word cloud for negative tweets related to poverty. All the frequently occurring words are having bigger fonts and are highlighted as bold whereas all the less frequent words all having smaller fonts and are not highlighted. Some of the frequently occurring words for negative poverty tweets include - hunger, economy, poor, politics, government, and inequality. Whereas, some of the less frequent words for negative tweets include - homeless, unemployment, ripping poor, and health-care poor.

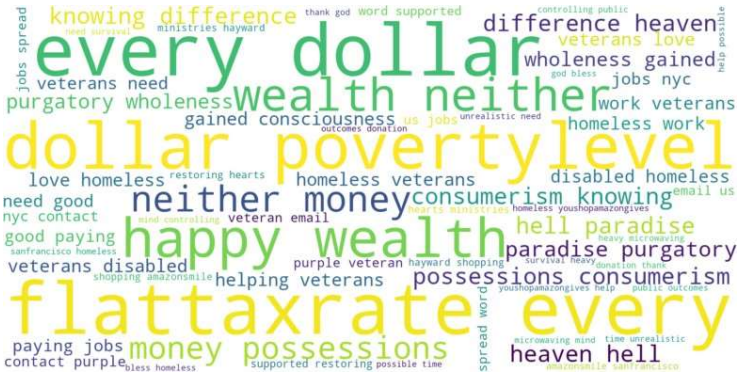

Fig. 51. WordCloud for positively classified poverty tweets

Figure 51 shows a word cloud for positive tweets related to poverty. All the frequently occurring words are having bigger fonts and are highlighted as bold whereas all the less frequent words all having smaller fonts and are not highlighted. Some of the frequently occurring words for positive poverty tweets include - dollar, happy, wealth, flattaxrate, and money possession. Whereas, some of the less frequent words for positive tweets include - paying jobs, good paying, veterans love, and disabled homeless.

15 CONCLUSION

This independent study analyzed the survey data of different families in Costa Rica for finding trends and insights in the poverty patterns and help social organizations to find the deserving families that require assistance. The study showed that income levels of family members are not the only sole indicators of poverty levels but education levels and number of people living in the house are highly correlated with the poverty levels. For this independent study, a detailed analysis of various features related to poverty such as size of the household, education levels of family members, housing material, monthly rent, sanitary conditions was performed and following conclusions were derived.

- Families living in better quality houses were not prone to extreme poverty conditions
- Family members having higher level of education or having greater than 18 years of average education had fewer chances of facing poverty conditions.
- Families having higher number of children and elders (more dependents) were vulnerable to poverty. Whereas, households having more adults (more non-dependents) usually were not prone to poverty.
- Similarly, families having more females (greater than 3) in the house were also affected by poverty.
- Families living in smaller houses (usually 1/0 bedrooms) also contributed to poverty levels. Also, the number of people living in the house were more (usually 8/9) in proportion to the size of the house (2/3 bedrooms), then those families were also vulnerable to poverty.
- Families owing tablets, computers, TVs and mobile phone did suffer from poverty.
- Also, families living in their own houses or could afford to stay in the houses having a very high monthly rent did not face poverty conditions.

Also, different data science techniques such as exploratory data analysis, correlation and regression analysis were performed for finding insights from the poverty dataset. Bar charts, box plots, scatter plots, and violin plots were used for exploratory data analysis to derive patterns from the data. Regression analysis helped to find trends (upward or downward) in the data and different conclusions were derived (all are discussed above in the bullets). Cross-correlation analysis assisted to eliminate redundant features which were highly correlated with other variables such as all the squared variables. Apart from this different feature reduction techniques such as PCA and t-SNE were used for reducing the redundant features from the dataset. Feature reduction reduced the over-fitting of the data due to curse of dimensionality problem. Several machine learning models such as Naive Bayes, Support Vector Machines, Random Forest, and K-Nearest Neighbors were implemented for predicting different poverty levels. Grid Search technique was used for fine tuning the hyper-parameters of machine learning models. All the machine learning models were evaluated using accuracy and f1-score on raw, under-sampled, and over-sampled datasets. Among all the machine learning models, random forest performed best and achieved an accuracy 91%of and f1-score of 0.75. In this independent study, a deep learning model was also implemented using Keras which achieved a prediction accuracy of 90%. Sentimental analysis of different poverty related tweets was performed using TextBlob API to understand people's sentiments and emotions related to poverty, homelessness and world hunger. A word cloud was also built on positively and negatively and negatively classified poverty tweets for knowing concerns of the public related to the poverty.

16 FUTURE WORK

The dataset used for this study was a survey data and was limited just to Costa Rica region. Hence, in future more data related to poverty and census could be collected for different countries. This could help governments to identity and reduce poverty not only just in Costa Rica but also worldwide. Apart from just text data, satellite images (about infrastructures, water bodies etc.) can also be included to analyze poverty [17]. It would also be interesting to explore how deep convolution neural network on the satellite images with different hyper-parameters tuning helps in analyzing the poverty.

REFERENCES

[1] [n. d.]. Costa Rican Household Poverty Level Prediction - Kaggle.com. ([n. d.]). Retrieved November 20, 2018 from https://www.kaggle.com/c/costa-rican-household-poverty-prediction

[2] [n. d.]. Matplotlib - Python plotting. ([n. d.]). Retrieved November 20, 2018 from https://matplotlib.org/

[3] [n. d.]. Model class API Keras Documentation. ([n. d.]). Retrieved November 25, 2018 from https://keras.io/models/model/

[4] Ahsan Anis. 2018. A Dummies Guide to Data Normalization. (Mar 2018). Retrieved November 22, 2018 from https://medium.com/@lahorekid/a-dummies-guide-to-data-normalization-for-neural-nets-ff1998116e75

[5] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python* (1st ed.). O'Reilly Media, Inc.

[6] Shelby Blitz. 2018. What Is Exploratory Data Analysis? l Sisense. (Jan 2018). Retrieved November 20, 2018 from https://www.sisense.com/blog/exploratory-data-analysis/

[7] Guest Blog. 2018. How to handle Imbalanced Classification Problems in machine learning? (Apr 2018). Retrieved November 22, 2018 from https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/

[8] Matt Brems. 2017. A One Stop Shop for Principal Component Analysis Towards Data Science. (Apr 2017). Retrieved November 24, 2018 from https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

[9] Adi Bronshtein. 2017. A ick Introduction to K Nearest Neighbors Algorithm. (Apr 2017). Retrieved November 23, 2018 from https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7

[10] Jason Brownlee. 2016. Develop Your First Neural Network in Python With Keras Step By Step. (May 2016). Retrieved November 25, 2018 from https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

[11] Ashley Crossman. [n. d.]. Correlation Analysis: Comparing Variables. ([n. d.]). Retrieved November 20, 2018 from https://www.thoughtco.com/what-is-correlation-analysis-3026696

[12] Luuk Derksen. 2016. Visualising high dimensional datasets using PCA and tSNE in Python. (Oct 2016). Retrieved November 24, 2018 from https://medium.com/@luckylwk/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b

[13] Rachna Devasthali. 2018. Coefficient of Determination (R-squared) Explained. (Apr 2018). Retrieved November 20, 2018 from https://towardsdatascience.com/coefficient-of-determination-r-squared-explained-db32700d924e

[14] Niklas Donges. 2018. The Random Forest Algorithm Towards Data Science. (Feb 2018). Retrieved November 25, 2018 from https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

[15] Melissa French. 2018. Presenting alitative Survey Data with Word Clouds. (Feb 2018). Retrieved November 25, 2018 from https://www.surveygizmo.com/resources/blog/qualitative-data-word-cloud/

[16] Rohith Gandhi. 2018. Support Vector Machine Introduction to Machine Learning Algorithms. (Jun 2018). Retrieved November 20, 2018 from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[17] Neal Jean, Marshall Burke, Michael Xie, W. Matthew Davis, David B. Lobell, and Stefano Ermon. 2016. Combining satellite imagery and machine learning to predict poverty. *Science* 353, 6301 (2016), 790–794. https://doi.org/10.1126/science.aaf7894 arXiv:http://science.sciencemag.org/content/353/6301/790.full.pdf

[18] Prateek Joshi. 2018. Comprehensive Hands on Guide to Twitter Sentiment Analysis with dataset and code. (Oct 2018). Retrieved November 20, 2018 from https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/

[19] Karol Karpinski and Asif Imran. [n. d.]. Building PMT-Based Social Registries — World Bank Group. ([n. d.]). Retrieved November 20, 2018 from https://olc.worldbank.org/content/building-pmt-based-social-registries

[20] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). arXiv:1412.6980 http://arxiv.org/abs/1412.6980

[21] Will Koehrsen. 2018. A Complete Introduction and Walkthrough. (Aug 2018). Retrieved November 20, 2018 from https://www.kaggle.com/willkoehrsen/a-complete-introduction-and-walkthrough?scriptVersionId=5196152

[22] William Koehrsen. 2018. A "Data Science for Good" Machine Learning Project Walk-Through in Python: Part Two. (Aug 2018). Retrieved November 20, 2018 from https://towardsdatascience.com/ a-data-science-for-good-machine-learning-project-walk-through-in-python-part-two-2773bd52daf0

[23] V. Kshirsagar, J. Wieczorek, S. Ramanathan, and R. Wells. 2017. Household poverty classification in data-scarce environments: a machine learning approach. *ArXiv e-prints* (Nov. 2017). arXiv:stat.ML/1711.06813

[24] G. Lemaitre, F. Nogueira, D. Oliveira, and C. Aridas. 2016. imblearn over sampling SMOTE. (2016). Retrieved November 22, 2018 from https://imbalanced-learn.org/en/stable/generated/imblearn.over sampling.SMOTE.html

[25] G. Lemaitre, F. Nogueira, D. Oliveira, and C. Aridas. 2016. imblearn under sampling RandomUnderSampler. (2016). Retrieved November 23, 2018 from https://imbalanced-learn.org/en/stable/generated/imblearn.under sampling. RandomUnderSampler.html

[26] Stevan Loria. 2018. TextBlob Simplified Text Processing. (Nov 2018). Retrieved November 26, 2018 from https: //textblob.readthedocs.io/en/dev/

[27] Author(s) Linden McBride and Austin Nichols. 2015. Improved poverty targeting through machine learning: An application to the USAID Poverty Assessment Tools. (2015). Retrieved November 20, 2018 from http://www.econthatmatters. com/wp-content/uploads/2015/01/improvedtargeting 21jan2015.pdf

[28] Ritchie Ng. 2018. Optimal Tuning Parameters. (Nov 2018). Retrieved November 22, 2018 from https://www.ritchieng. com/machine-learning-efficiently-search-tuning-param/

[29] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2825–2830. http://dl.acm.org/citation.cfm?id=1953048.2078195

[30] NATAA PLULIKOVffi. 2015. POVERTY ANALYSIS USING MACHINE LEARNING METHODS - uniba.sk. (Oct 2015). Retrieved November 20, 2018 from http://www.iam.fmph.uniba.sk/institute/stehlikova/BC/2016-plulikova.pdf

[31] Sunil Ray. 2014. 7 Types of Regression Techniques you should know. (Aug 2014). Retrieved November 20, 2018 from https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/

[32] Sunil Ray, Business Analytics, and Intelligence. 2017. Understanding Support Vector Machine algorithm from examples (along with code). (Sep 2017). Retrieved November 23, 2018 from https://www.analyticsvidhya.com/blog/2017/09/ understaing-support-vector-machine-example-code/

[33] Joshua Roesslein. [n. d.]. Tweepy API Reference. ([n. d.]). Retrieved November 26, 2018 from http://docs.tweepy.org/ en/v3.5.0/api.html

[34] saedsayad. [n. d.]. Naive Bayesian. ([n. d.]). https://www.saedsayad.com/naive_bayesian.htm

[35] Saurabh. 2017. Comprehensive Guide on tSNE algorithm with implementation in R and Python. (Mar 2017). Retrieved November 23, 2018 from https://www.analyticsvidhya.com/blog/2017/01/t-sne-implementation-r-python/

[36] Tavish Srivastava. 2018. Introduction to KNN, K Nearest Neighbors : Simplified. (Mar 2018). Retrieved November 23, 2018 from https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

[37] L.J.P. van der Maaten and G.E. Hinton. 2008. Visualizing High-Dimensional Data Using t-SNE. (2008).

[38] Michael Waskom. 2012. seaborn: statistical data visualization. (2012). Retrieved November 20, 2018 from https: //seaborn.pydata.org/