Income Level Prediction Using a Variety of Machine Learning Algorithms

Tushar Kataria

***Abstract*—Poverty is one of the most important problems in the world. With more than 90% of wealth accumulated with just top 1% of the humans in the world. We should design policies which can help divide the wealth in more equitable way and reduce poverty. Currently 689 million people live in poverty across the world. Even in a wealthy country like USA, approx 10 percent of people live in poverty. New policies tackling poverty should be based on data, which is a very hard to obtain because collecting data is a time consuming and costly task. Therefore we need sophisticated models which can predict statiscs on earning of population of large counties, cities, states, countries etc using parameters which might be easily available like age, gender, educational qualification etc. Creating these models will help the policy makers create new laws, remove non-working laws or initiatives and also help in evaluation & tracking of new initiatives. In this analysis we use different machine learning algorithms(Decision Tree, Random Forest, deep learning models etc), a variety of data pre-processing mechanisms(Mean, Median, one hot encoding) to find out which algorithms is best suited for analysis and tracking of income prediction data. All the code and results are added to git directory. Link to Projecy Github Folder**

I. Introduction

Nowadays machine learning is everywhere emails, social media, texting, calls , shopping on amazon, walmart, financial markets and lot more. Due to the robust nature of machine learning algorithms they are applied in almost every domain of science and technology and recent advances in deep learning models have pushed the boundaries even further.

In this paper we are looking at the problem of predicting if certain individual's income is less than 50k$ or not. The training dataset provided has the following attributes:-

- age :- Continous varible
- workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
- education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool
- education-num: continuous.
- marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
- occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces
- relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
- race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
- sex: Female, Male
- capital-gain: continuous.
- capital-loss: continuous.
- hours-per-week: continuous.
- native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands

After analysis of data it was found that there are missing values for several data points in "workclass", "Occupation" and "native.country". The training data is a combination of both categorical and numerical data so will require some data preprocessing before any machine learning algorithms can be applied. The training data is also heavily skewed for some attributes like "native.country", where the percent of data points with "United-states' is almost 90%. The aim of the project is to find an algorithm and data preprocessing technique which has the highest Area Under ROC curve for Testing data.

Section II Gives a brief explaintation of all the data preprocessing algorithms tried for both categorical and numerical attributes. Section III lists and breifly explain all the algorithms tried. Section IV decribes the experimental setup followed for training all the algorithms. Section V is a discussion on the results. Section VI is conculsion and future work possible.

II. Data Pre Processing Methods

Data preprocessing is very common and sometimes very important for many datasets. Models trained on unclean data don't generalize well to unseen data. Therefore its imperative that we clean the data however we can and then apply machine learning algorithms to find a good learned model. Some Data preprocessing methods were discussed in lecture [4], but here pre-implemented encoder will be tried on the dataset. As explained above, this data consist of both neumerical and categorical data, so there will be data preprocessing for both of these features.

A. Datapreprocessing for Categorical Data

For transforming categorical data to numerical data a scikit-learn category encoder library was used. 15 different categorical transformers from that library were tried on the dataset. Below is a a short explaination of a few category encoders [3], other definations can be found on the Webpage Here.

- **Ordinal Encoder** :- This encoder simply just assigns a number between 0 and number of unique attributes of that feature. So for example education dataset has 16 unique values, oridinal will give each value a number between 0 and 15.
- **One Hot Encoder** :- This encoder assign a unique binary code to each value of the categorical feature. so for example education has 16 values, one hot encoder will turn these 16 values to 16 features assign 1 to the current value and 0 to all others. This encoding increase the size of features.
- **BackwardDifferenceEncoder** :- "In backward difference encoding, the mean of the dependent variable for a level is compared with the mean of the dependent variable for the prior level." Explaination Taken from here.
- **SumEncoder**
- **HashingEncoder** :- Transfomation to high dimentional space of integers.
- **Binary Encoder** :- In this encoding scheme each category is first assigned a numerical value and then that numerical value is converted to a binary number.
- **Target Encoder** :- This is a Bayesian Encoding Technique. For Target Encoding we calculate the mean of the target variable for each category and replace the category variable with the mean value.
- **MEstimate Encoder**:- This is a simplied version of Target Encoder. More Detials on the webpage.
- **Polynomial Encoder**
- **Leave One Out Encoder** :- This is also a simplied version of Target Encoder. More Detials on the webpage.
- **James SteinEncoder**
- **Helmert Encoder**
- **Generalized linear mixed model Encoder**
- **Count Encoder** :- This encoding replaces the names of the categorical features with the appearance counts.
- **Weight of Evidence coding**

B. Datapreprocessing for Numerical Data

There are many datapreprocessing techniques for Numerical data transformation. The ones tried in the study are :-

- **Standard Scaling** :- This coding maps the numerical features to a 0 mean and unit variance.
- **Min Max Scaling** :- Scaling each feature by maping it to a range of [0,1].
- **Max Absolute Scaling** :- Scaling Each feature by it's maximum absolute value.
- **Robust Scaling** :- This Scaling is robust to outliers and scales the data using Inter Quantile Range.
- **Quantile Transformation** :- Maps features to uniform or normal distribution using quantile information. More robust to outliers.
- **Power Transformation**:- this is also a 0 mean, unit variance normalization but uses Box-Cox transform or the Yeo-Johnson transform for calculating output features.
- **Log Transformation**
- **Normalization** :- Normalizes each sample to unit norm.
- **Polynomial Transfomation** :- Polynomial feature transformation of degree 2.

All Combination of these preprocessing techqniues were tried for different algorithms.

III. Machine Learning Algorithms

Several Machine Learning Algorithms were used. Initially a lot of classfiers were trained but after realizing the mistake started working with regrssion algorithms. Most of the algorithms implementation are used from sklearn Library. For Many algorithms ROC-AUC score were not noted down because they didn't perform better than curent kaggle submission. But for many of them the results are shown in Section V. List of the algorithms Tried are given below:-

- **Decision Trees Regressor** :- An extension of decision tree's to regression problem.
- **Random Forest Regressor** :- Random Forest based on Decision Tree regressor as the base regressor.
- **Gradient Boosting Regressor** :- This is a family of algorithms which work similar to adaboost but have an differential loss function. These family of algorithms are better than random forest.

- **Histogram Gradient Boosting Regressor** [2]:- This is an optimized implementation of Gradient boosting algorithm.
- **Support Vector Machine Regression** :-Linear, polynomial and RBF kernel.
- **Orthogonal Matching pursuit** :- This is a projection based learning algorithm onto to span of dictionaries learned from data.
- **Gaussian Process Regressor**
- **Linear Regression** :- Least squared fitting to data.
- **Stochastic Gradient Descent Regressor** :- Linear model fitted by minimizing a regularized loss by using SGD.
- **Bags of regressors**:- 3 bags of regressor were tried. 30 bags of Decision Trees, 20 bags of Gradient Boosting Regressor and 20 bags of Histogram Gradient Regressor.
- **Adaboost Regressor** :- Base regressors used with adaboost regressors were Decision Trees of depth 3, Gradient boosting Regressor of depth 3. Depth was fixed so as to learn weak learners.Even Tried Linear Regression because Linear regression didn't perform well on the dataset, so they might be viewed as weeklearners.
- **XGBoost**:- This is another optimized version of Gradient boosting methods. [5].
- **Neural Networks** :- Linear layers of different depths and different non-linearities like ReLU, Tanh, etc.
- **Elastic-Net** :- This is a linear regression model which uses both L1 and L2 Regularization for optimization.

A. *Feature Selection*

IV. Experimental Setup

This experimental setup is consistent across all the training algorithms tried. The training data is split into different sized training and validation sets using strafied sampling so as to maintain the label imbalance in both training and validation sets. From full training data five independent sets of training and validation sets were created with train-validation split of 50-50, 60-40, 70-30, 80-20 and 90-10 using stratified sampling. Some of the algorithms have a tendency to overfit the training data, I think splitting the data in multiple sets cements the validitity of an algorithms accuracy on unseen data.

V. Results

This is a kaggle competition so results are only Avialable for 30% of the test data. Instead of reporting Testing data numbers, for comparison of algorithms, Validation ROC numbers are reported. Testing Number will be report in a Single table at the end in Table XIV . Generally the Kaggle Testing ROC is less than Validation ROC but the values are generally very close.

Table I shows the results for Decision Tree Regressor. The Highest Validation ROC was seen for 90-10 Split. Table II shows the results for a bag of 30 decision trees regressor. Comparing the results of bags of trees with normal decision tree regressor, there isn't much difference in terms of ROC value. Best Numerical encoder for decision tree regressor is polynomial or maximum absolute regressor but no clear best category encoder. Table III shows the results with Adaboost Regressor when using Decision Tree Regressor of depth 3 for each base regressor. Comparing ROC of Adaboost with Bags of regressor and normal decision tree regressor, there is still no big difference between the Validation ROC values.
Table IV shows the Validation ROC with decision tree as base regressor. Random Forest regressor is working much better than Decision tree regressor, bag of decision tree regressor and Adaboost.
Table V shows the Validation ROC when using Gradient Boost Regressor. Validation ROC is quite improved when compared with almost all algorithms except histogram gradient boosting. Bags of Gradient Boosting performs slighly better than simple Gradient boosting Regressor. Adaboost VII with base regressor works slightly better than Gradient boosting but is similar in Validation ROC performance of Bag of Gradient boosting.
Histogram Gradient boosting performs better than gradient boosting as shown in Table XI. Bags of Histogram Gradient boosting regressor perform slightly better than simple histogram gradient boosting regression as shown in Table XII.
Validation ROC values for the Linear Regression is listed in Table IX . Linear Regression doesn't perform better than any other algorithm, it is similar in performance to Decision Tree regressor and also SGD regressor shown in table X.

A. *Observations*

The following observation were made from the experiments and the data collected:-

1) Gradient Boosting and their optimized version outperform any other machine learning algorithms tried on the dataset. The Highest Validation ROC value were observed for these family of algorithm. Though gradient boosting algorithms have a high probability of overfitting, Training ROC using these algorithms (approximately 0.94-0.95) was not too far.
2) Linear models perform very poorly on the dataset even with non-linear feature mapping.
3) Non-Linear models like Decision tree regressor perform better than Linear models.

4) Decision tree regressor with smaller depth tend to perform better on validation set compared to fully expanded tree. This same observation was also made for gradient boosting algorithm. Max-depth being a hyperparamter in both. Generally for both Gradient boosting and Decision tree, max-depth of about 7-10 was giving best results.
5) Bags of regressor, doesn't make that much of a difference in validation ROC. For all three algorithms tried for bags of regressor, the final validation and testing ROC was not that far from original algorithms performance. Similar observation as made for Adaboost algorithm.
6) Neural network models performed fairly well on the datset. But for the dataset, wasn't able to train a model which can outperform gradient boosting. The Maximum Validation ROC using neural network was close to 0.92. Both ReLU and Tanh Non linearity performed similarly on this dataset.
7) No numerical or categorical encoding scheme stands out as giving best results across all algorithms. Different algorithms acheive optimums for different data encoders.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9006	Polynomial	Polynomial
60-40	0.9008	Helmert	Polynomial
70-30	0.9030	JamesStein	Polynomial
80-20	0.9068	Hashing	NoScaling
90-10	**0.9077**	Binary	MaxAbsScaler

TABLE I
DECISION TREE REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9011	JamesStein	QuantileTransformer
60-40	0.9026	Count	MinMaxScaler
70-30	0.9031	Polynomial	MinMaxScaler
80-20	0.9089	LeaveOneOut	RobustScaler
90-10	**0.9117**	Binary	logNormal

TABLE II
BAG OF 30 DECISION TREE REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9077	Count	NoScaling
60-40	0.9054	Count	Standard
70-30	0.9064	Polynomial	Polynomial
80-20	0.9071	JamesStein	NoScaling
90-10	**0.9086**	Helmert	NoScaling

TABLE III
ADABOOST WITH DECISION TREE REGRESSOR OF DEPTH 3 BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9101	LeaveOneOut	MaxAbsScaler
60-40	0.9136	Count	NoScaling
70-30	0.9121	Polynomial	QuantileTransformer
80-20	0.9147	LeaveOneOut	logNormal
90-10	**0.9174**	OneHot	logNormal

TABLE IV
RANDOM FOREST REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9216	OneHot	NoScaling
60-40	0.9233	Helmert	NoScaling
70-30	0.9279	LeaveOneOut	Polynomial
80-20	**0.9295**	Polynomial	NoScaling
90-10	0.9276	GLMMEncoder	MaxAbsScaler

TABLE V
GRADIENT BOOSTING REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

VI. CONCLUSION AND FUTURE WORK

Family of Gradient boosting algorithms are the best performing on the dataset given. Different data encoding schemes give optimized regressors for different algorithms. There is no consistent data encoding scheme which stands out for all algorithms. In this study only only feature selection was tried(K-best in sklearn), but that didn't give better results in terms of ROC values. Also removing data with unknown points from training dataset didn't improve the Validation ROC. For future work, different feature selection techniques can be tried out. As it was noted in Section I that data is imbalanced, this can also be countered by undersampling or oversampling using SMOTE [1]. Usage of SMOTE was tried but not explored fully.

REFERENCES

[1] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
[2] Zeyu Feng, Chang Xu, and Dacheng Tao. Historical gradient boosting machine. In *GCAI*, pages 68–80, 2018.
[3] Will McGinnis. Category encoder scikit library. In *Category Encoder 2.2.2*, pages 68–80, 2016.
[4] ShandianZhe. Lecture notes, slides and video lectures. In *CS5350/6350 Machine Learning, University of Utah*, page 1, Jan-Mar 2021.
[5] xgboost developers. Xgboost. In *2020*, pages 68–80, 2020.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9260	GLMM	Standard
60-40	0.9278	Hashing	NoScaling
70-30	0.9279	Polynomial	MaxAbsScaler
80-20	**0.9357**	Hashing	NoScaling
90-10	0.9334	MEstimate	MinMaxScaler

TABLE VI
BAG OF 20 GRADIENT BOOSTING REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9268	JamesStein	MaxAbsScaler
60-40	0.9261	Target	RobustScaler
70-30	0.9305	LeaveOneOut	logNormal
80-20	**0.9320**	Polynomial	NoScaling
90-10	**0.9320**	Polynomial	NoScaling

TABLE VII
ADABOOST WITH GRADIENT BOOSTING REGRESSOR OF DEPTH 3 BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.8991	Helmert	MaxAbsScaler
60-40	0.8982	Polynomial	MinMaxScaler
70-30	0.9033	Polynomial	MaxAbsScaler
80-20	0.8979	Sum	Standard
90-10	**0.9017**	JamesStein	MaxAbsScaler

TABLE VIII
SVM-RBF BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9099	GLMM	Polynomial
60-40	0.9070	Hashing	Polynomial
70-30	0.9095	Polynomial	Polynomial
80-20	0.907	Count	Polynomial
90-10	**0.9106**	MEstimate	Polynomial

TABLE IX
LINEAR REGRESSION BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.8991	Hashing	QuantileTransformer
60-40	0.8979	Hashing	QuantileTransformer
70-30	0.9000	Polynomial	QuantileTransformer
80-20	0.8999	Count	QuantileTransformer
90-10	**0.9001**	Binary	QuantileTransformer

TABLE X
SGD REGRESSION BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9262	JamesSteinGLMM	MaxAbsScaler
60-40	0.9292	JamesStein	NoScaling
70-30	0.9299	Target	NoScaling
80-20	**0.9341**	Hashing	NoScaling
90-10	0.9324	MEstimate	RobustScaler

TABLE XI
HISTOGRAM GRADIENT BOOSTING REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.9288	Hashing	NoScaling
60-40	0.9293	JamesStein	NoScaling
70-30	0.9292	Binary	NoScaling
80-20	**0.9357**	Hashing	NoScaling
90-10	0.9337	BackwardDifference	NoScaling

TABLE XII
20 BAGS OF HISTOGRAM GRADIENT BOOSTING REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Split	Validation ROC	Cate Encoder	Numerical Encoder
50-50	0.8837	CatBoost	Polynomial
60-40	**0.8897**	CatBoost	Polynomial
70-30	0.8860	CatBoost	Polynomial
80-20	0.8739	CatBoost	Polynomial
90-10	0.8694	CatBoost	Polynomial

TABLE XIII
ELASTIC NET REGRESSOR BEST VALIDATION ROC'S ACHEIVED, BEST CATERGORY ENCODER AND BEST NUMERICAL ENCODER.

Algorithm	Best Testing ROC on 30% data
Decision Trees	0.90948
Random Forest	0.91056
Gradient Boosting	0.91891
Hist Gradient Boosting	**0.92148**
30 Bags of Decision Trees	0.89407
20 Bags of Gradeint Boosting	**0.92139**
20 Bags of Hist Gradient Boosting	**0.92139**
Adaboost with Gradient Boosting	0.92055
Neural Networks	0.90845
XGD boost	0.91941

TABLE XIV
TESTING ROC VALUES FOR DIFFERENT ALGORITHMS ON 30% OF TESTING DATA AS VISIBLE ON KAGGLE.