Project Report

Predicting a newlander’s future tax bracket with stacked ensembles of Gradient Boost and Naive Bayes as base estimators

Curricular Unit: **Machine Learning**

Group Members: **Daniel Correia (20200665),**

Gonçalo Reis (20200650),

Joana Rafael (20200588),

Ricardo Santos (m20200620)

Date: 27.12.2020

Version: v1.0

Abstract

The current work uses a diverse set of classification machine learning algorithms in order to derive a predictive model with the capacity to correctly classify whether a citizen's income is above or below the average income of the general populace.

The model was derived from a dataset originally with 22,400 observations and 15 attirbutes. The model was later split into training-validation (70%) and test sets (30%). The models were trained with RepeatStratifiedKFold cross-validation and their performance on the test set was verified through F1-Score, ROC Curve and Precision-Recall.

The authors report that a stacking ensemble algorithm, using Gradient Boosting and Naive Bayes as its base estimators and Logistic Regression as its meta-estimator, is the most efficient predictor for the binary outcome of this classification problem. Further optimization techniques resulted in slight improvements of Recall which resulted in a slightly lower F1-Score but higher AUC on the ROC curve (0.919 from 0.9173) and AP on the Precision-Recall Curve (0.8084 from 0.8024) respectively, which indicates that the model's final version has better capabilities in distinguishing classes.

The final model's F1-score on the split test set was 0.866 . The same model scored 0.86501 on the 30% publicly available results of the Newland government's test set. The similarity in results indicate robustness of the models predictive power.

I. Introduction

In 2039, humanity discovered a new planet compatible with known life inside The Milky Way. The discovery brought hope into the life of a generation who felt nothing but hopelessness: The Paris Agreement of 2015 had been the closest to a global climate strategy humanity had reached, leaders around the globe had failed again and again to reach a more meaningful agreement since it and decades of unfeathered industrial activity had brought the planet past the point of no return. Newland, as the planet was later called, meant, above all, the possibility of a new beginning for humanity without the consequences of decades of contempt for environmental sustainability. In 2046, the first 40 000 new settlers arrived successfully at the new planet.

Two years later, 100 more ships were launched to Newland. The expected drastic increase in population led the Newland government to consider the application of a binary tax rate: people whose income is above the average income will be taxed 30% and the remaining citizens are to be taxed 15%. The effectiveness of progressive tax rates has been a popular topic of discussion between economic academics (Guillaud et al., 2020), the incentives for tax evasion of high-income earners seem common knowledge (Alstadsæter et al., 2019). It is, then, important to improve the government's ability to correctly predict the income of its citizens.

In a preliminary study, the Newland government sampled data on 32,500 adults and their incomes. With that data, the government challenged different data science teams to develop a predictive model for the tax rate each citizen should pay.

Predictive models based on machine learning (ML) algorithms are effective in predicting income from e.g. census data (Chakrabarty and Biswas, 2018). The use of ML has increased significantly since the beginning of the 21st century. Regardless of the field of expertise, researchers and developers have realized the potential of showing a machine the desired input-output behavior and allow it to make predictions based on previous observations (Jordan and Mitchell, 2015).

Among the most popular frameworks for ML projects is the Cross-Industry Standard Process for data mining (CRISP-DM) which is an iterative and standardized process whose 6 fundamental steps cover the requirements of a successful ML project: *Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment* (Chapman et al., 2000). The current document reflects the author's CRISP-DM approach to developing a predictive ML model for the income of the citizens of Newland.

II. Background

In this section, the authors will explore concepts, techniques, and assumptions that have either been only briefly covered, or not at all, in Machine Learning classes. In the interest of clarity, the explanations will be separated into 3 main topics: i) **fundamental assumptions about the data**, ii) **data engineering techniques, particularly upsampling and downsampling** and **iii) algorithms.**

Fundamental assumptions about the data

Before going into detail on the adopted methodology, it is relevant to lay out the main underlying assumption when it comes to working with and making predictions from data. To be able to predict future outcomes (or outcomes for unlabeled data) based on the past (or labeled data), it is necessary to assume that: i) there is some *fundamental statistical law (or pattern)* that is shared between the past observations and the future observations and ii) the observations are independent of one another. These assumptions ensure that the data is independent and identically distributed (iid), which, in turn, guarantees that it is possible to use labeled data to make generalized claims about unlabeled data (Nouretdinov et al., 2001). The iid assumption has been used in financial risk assessment (Feng et al., 2008) before and will be adopted in the current work. However, it is relevant to highlight that there have been efforts to develop ML-based solutions (e.g. recommender systems) whose focus is to ignore iid assumption (Hu, 2018).

Data engineering: Upsampling and Undersampling

The Newland government's dataset is not balanced (i.e. the dataset's possible targets are split 70-30, with over-representation of citizens that are lower-income earners). The issues that may arise from unbalanced datasets and how they can be dealt with have been previously discussed and reviewed (S. Kotsiantis et al., 2006). When necessary, the authors relied on upsampling techniques to address the imbalance issue.

Upsampling (particularly random oversampling) is a preprocessing technique that randomly replicates the class with less expression until the desired proportion between classes is met. The process does output a balanced dataset (wherein the members of each class are evenly represented). However, it also comes with its drawbacks. In particular, as the technique replicates data and appends it to the original dataset, the appearance of multiple repeated instances may lead to overfitting of training data and less overall predictive power (Haibo He and Garcia, 2009).

Undersampling represents the opposite process. Random oversampling removes rows from the over-represented class in the original dataset which may lead to the model being unable to distinguish potentially important relationships that are specific to the majority class.

Algorithms: Multinomial Naive Bayes Classifier

At the most fundamental level, supervised ML uses data to make inferences about some unobserved relationship, use said inferences to make predictions, and assess what adjustments to make to improve its predictive power. Predictions of the unknown have uncertainty associated with it. Therefore, it is possible to look at an ML problem through the eyes of probability theory, which is generally referred to as Bayesian learning which can be applied to both regression and classification problems (Ghahramani, 2015).

Bayesian classification was introduced and covered previously in machine learning classes. In short, the assignment of a class is made according to the probability of belonging to that class. The general Bayesian approach to model optimization in classification problems is also simple: the optimal classifier (which is the Bayes classifier) is the classifier that minimizes the probability of making a wrong prediction. That probability can be calculated in the following way:

$$p(f(X) \neq Y) = \sum_{i=i}^{n} p(Y = yi \mid X = xi). E(f(xi) \neq yi)$$

For each value of feature x_i in X and target y_i, we can say that the probability of making an incorrect prediction is the sum of the probabilities of y_i given x_i when the expected value of $f(x_i)$ is different from

y_i. Under the iid assumption, the classifier that minimizes that probability is the one that maximizes the probability of x_i given y_i's true value:

$$p(X = xi \mid Y = yi)$$

However, the true values of these probabilities are unknown and follow an unknown distribution. Naive Bayes addresses both of these issues: i) the conditional probabilities of x_i given y_i are easier to compute by assuming that, given $y_{i,}$ the features are independent of one another and ii) the unknown distribution that the data follows can be approximated to a known distribution (Domingos and Pazzani, 1997). The Multinomial Naive Bayes Classifier is a classifier that considers that the data follow a multinomial distribution, which is a generalization of the Bernoulli distribution commonly used in text classification (Kibriya et al., 2004).

Algorithms: Receiver Optimization Characteristic (ROC) curves and Precision-Recall

ROC was first used to refer to the ability of a World War II radar operator to correctly classify a signal as an object or noise and the concept has since been applied to other fields of expertise (Fan et al., 2006). In a binary classification problem (such as the one presented by the Newland government) the ROC curve plots the fraction of zeros that are misclassified as ones (x-axis) against the fraction of ones that were correctly assigned by the classifier (y-axis). The higher is the value of the area under the curve (AUC), the better the classifier can correctly distinguish between classes.

Different authors have argued for (Provost et al., 1998) and against the efficacy of the ROC curve (Drummond and Holte, 2004). Among the alternatives to the ROC curve is Precision-Recall, a method discussed and used in class (Davis and Goadrich, 2006).

III. Methodology

The current section will follow focus on the methods and transformations that allowed the authors to go from the raw dataset to the final prediction. The process relied on the logical steps of the CRISP-DM framework. Therefore, the current section will be split into the framework's 6 main steps: **Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment.** Due to its experimental and iterative nature, CRISP-DM is a dynamic and fluid approach where projects go back and forth from step to step. Thus, it is not uncommon for projects on a more advanced phase (e.g. Model Evaluation) to be sent to a previous phase (e.g. Data Preparation) for additional transformation. In the interest of clarity, all transformations on the data pre-model selection will be displayed on the Data Preparation section and all transformations made post model selection will be placed on the Modeling or Evaluation sections.

III.1. Business Understanding

The Newland government provided all challenging teams with a training dataset (with 15 total attributes and 22,400 subjects) and a Test dataset (with 14 attributes and 10,100 subjects). Both datasets share 14 attributes with the remaining attribute of the Training data – *Income* – representing the targets. The targets are represented by a binary classification key: if a subject's income is under the average income of the general population, its value will be 0 and the citizen will be charged 15% in taxes. If the subject's income is higher than the average, the value on the *Income* variable will be 1 and the subject's tax rate will be 30%.

The 14 attributes initially provided by the Newland government include *Occupation, Marital Status*, *Weekly Hours Worked*, whether the considered subject paid (or was paid) to be present on the expedition. In the next steps, these features will be interpreted and transformed to assess their predictive power over the target.

III.2. Data Understanding

A preliminary overview of the data's properties was performed. The exploratory data analysis revealed that 5 out of the 14 initial features (*Citizen ID*, *Ticket Price*, *Money Received*, *Working Hours per Week,* and *Years of Education*) are numeric variables the remaining 9 are categorical/ordinal features.

The authors relied on the appropriate methods of the *pandas* library to identify missing values, potential outliers, and uncover the statistical properties and attribute behavior of each feature. The dataset did not showcase blatant errors in data entry or errors.

III.3. Data preparation

The following manipulations of the original dataset were made as a consequence of the immediate observations made in the previous phases:

i) *Citizen_ID* was set as the unique identifier of each citizen, with variable *Name* being converted into *Gender* (rows with *Mr.* were converted to 1, *Mrs.* and *Miss* to 0),

ii)The *Date of Birthday* attribute was converted into *Age* by subtracting the year of birth from the year the expedition (2048),

iii) Entries with *?* and *nan* were subjected to replaced by np.nan. Missing values in the *Base Area* or *Employment Sector* were replaced by their corresponding mode. Missing values on *Role* were replaced with *Unkown.*

The following manipulations were performed before model selection but do not result from immediate or self-evident observation. They result instead of heuristics adopted by the authors:

i) The authors agreed upon a static 75% threshold for the concentration of a feature. All features where a particular value is represented in, at least, 75% of the observations. The adoption of this heuristic led to the discard of two features: *Base Area* and *Native Continent*.

ii)All remaining categorical features were subjected to *One-Hot Encoding*. In essence, n-1 out of n possible values of each categorical feature were converted into a binary dummy feature.

iii) The newly generated feature *Unemployed* was discarded due to under-representation: 19 observations in the training dataset (0.08%).

iv) Upon investigation of the distribution of our numerical variables the authors concluded that the outliers seem to be natural variations of the data distributions, rather than measurement errors, data entry errors or not part of the population in study. While these are oddballs, they accurately reflect the potential surprises and uncertainty inherent in ages, years of education or working hours per week, money received or ticket price. However, for the latest two variables, having such a wide range or extreme values can be detrimental to the performance of our model, so the authors applied a log transformation to them, as an attempt to reduce the variance between these datapoints (although evidence for different performance in the final model, with or without log application was not found). Although the authors did not find evidence for a better perfoming model after removal of these outliers, even if these unusual observations influence the model's performance, the authors made the decision of keeping them, as it can be bad practice to remove data points just for the sake of simply producing a better fitting model or statistically significant results.

v)All numeric values were scaled to values between 0 and 1 with Sci-kit learn's MinMaxScaler method and had their Pearson correlation index analyzed for feature selection purposes (Figure 1). The low correlation values between features led the authors to not discard any additional numerical features.

vi) For feature selection, the authors obtained the 10 most significant features under the Chi-square test and intersected them with the 10 most significant features of the Mutual Information Coefficient (MIC) testing method (Jiawei Han et al., 2011). The results for both tests are shown in Figure 2. All other categorical features were discarded for model selection steps.

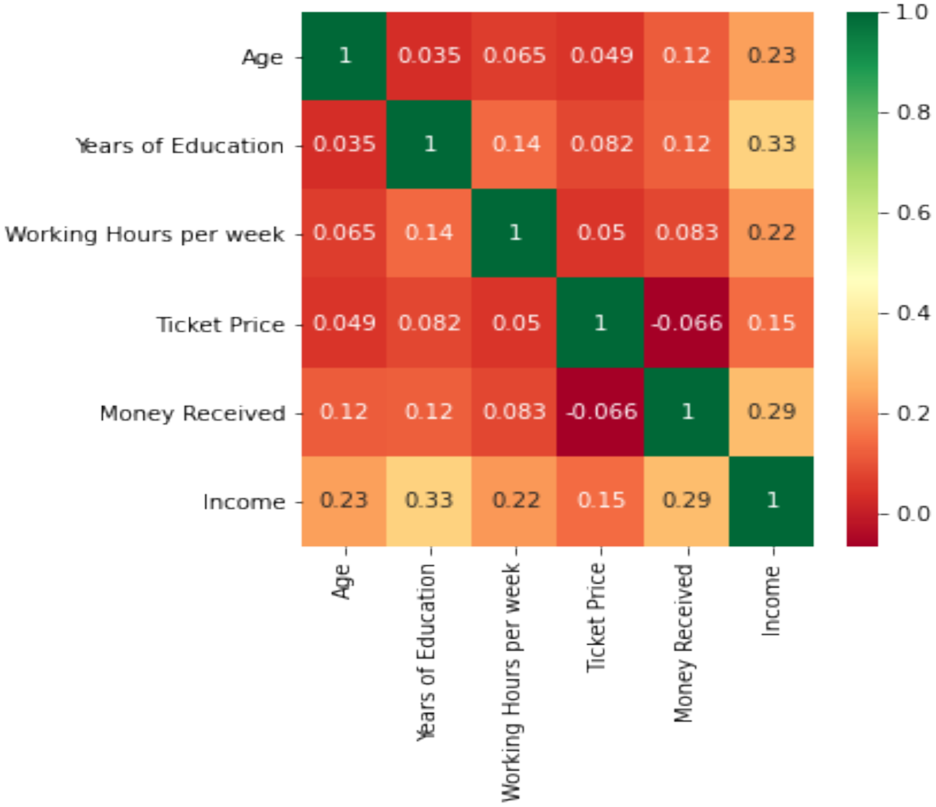

Figure 1: Pearson correlation matrix of numerical features. The low values reflect that there is low linear correlation between features.

At the time of transition to the model assessment phase, the features that remained were: *Age*, *Money Received*, *Ticket Price,* and *Working Hours per Week* as numerical features; *x1_Married*, *x2_Spouse*, *x3_Masters*, *x1_Married*, *x2_Other Family*, *x2_Wife*, *x1_Single*, *x2_Children*, *x4_Professor*, and *x4_Management*. The transformed dataset was split (random state 15) into training/validation data (70%) and notebook testing data (30%).

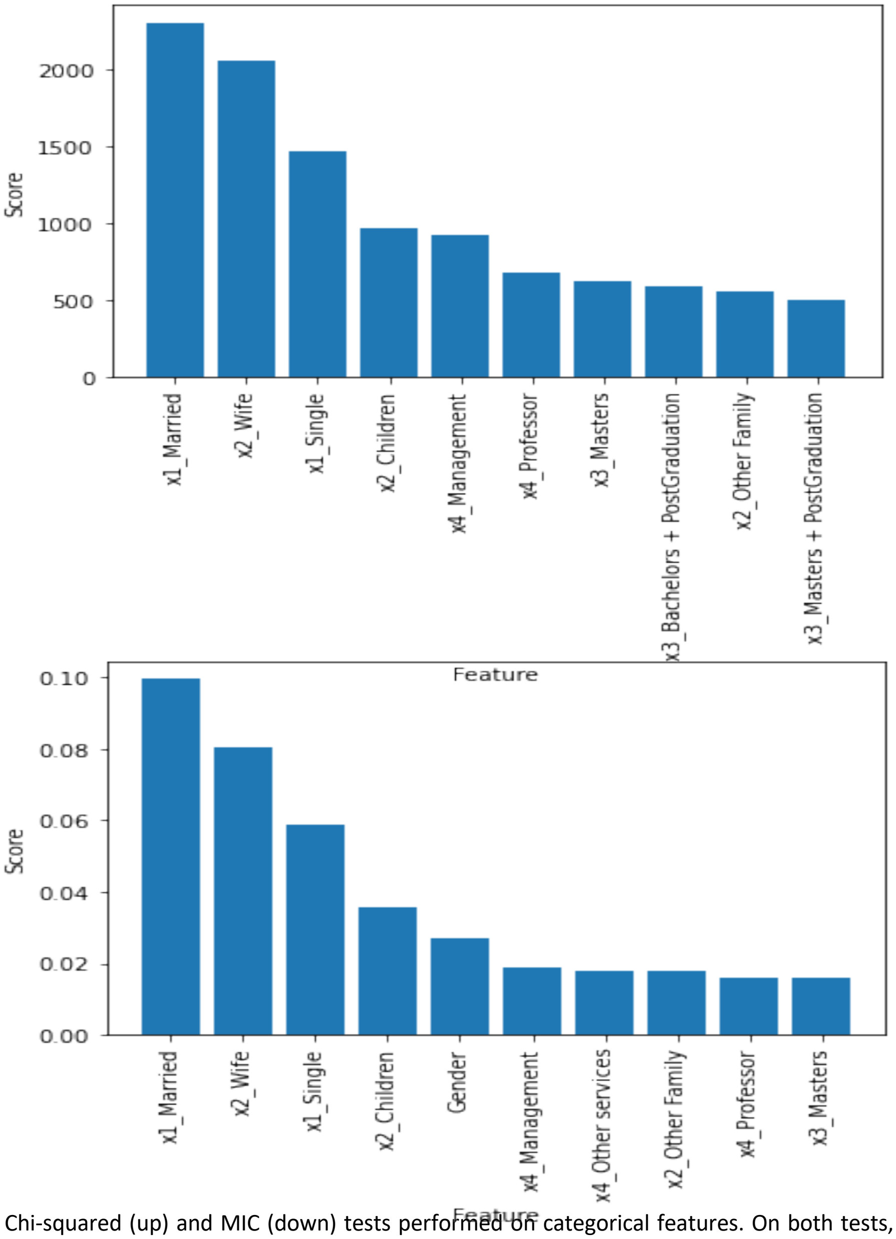

Figure 2: Chi-squared (up) and MIC (down) tests performed on categorical features. On both tests, the most significant features were X2_Married, x2_Wife, x1_Single and x2_Children, with differences arising at 5th most important feature.

In this phase, the authors focused on training and evaluating the performance of several learning algorithms (standalone models) considered to be adequate prospects by the authors: K-Nearest Neighbors (KNN), Logistic Regression (LR), Gaussian (NB), Binomial (BNB) and Multimonial (MNB) Naive Bayes Classifiers, Neural Networks (NN), Decision Trees (DT) and Support Vector Machines (SVM). Model training had cross-validation with Stratified Repeated K Fold with 5 splits and 5 repeats and, after training, the model was fed the sampled test set from that resulted from the initial split. Model adequacy was evaluated through analysis of the results on the created Notebook testing set on 3 different parameters: F-1 score (micro), Precision-Recall, and the Roc Curve. For all models mentioned, the variable income was set as the Target while the remaining variables were set as features.

The results obtained led to additional tests with ensemble methods that take NNs or DTs as estimators (Bagging NNs, Random Forests, AdaBoost and Gradient Boost). The highest performing algorithm was used as benchmark for the following 17 possible stack combinations of estimators (#1 (*RF*, *NN*); #2(*RF*, *NB*); #3(*NN*, *NB*); #4(*RF*, *SVM*); #5(*NB*, *SVM*), #6(*GBoost*, *NB*); #7(*GBoost*, *SVM*); #8 (*Gboost*, *NN*); #9 (*MNB*, *NB*, *DT*); #10 (*GBoost*, *LR*); #11 (*GBoost*, *AdaBoost*); #12 (*GBoost*, *AdaBoost*, *NB*); #13 (*RF*, *NB*, *SVM*); #14 (*DT*, *NB*, *SVM*), #15 (*LR*, *SVM*); #16 (*LR*, *NB*) and #17 (*NB*, *LR*, *SVM*)). The performance of the stacked models was evaluated by the same metrics as the standalone models. On this step, the report will only highlight results of models that performed better than the benchmarked algorithm.

III.5. Evaluation

After assessing model adequacy and selecting which algorithms to keep optimizing, the authors performed additional feature engineering on the data. At this stage, editions to the dataset were performed iteratively and continuously assessed. Editions were validated if and only if they resulted in improved model performance (measured by the F1-score, the ROC-Curve and Precision-Recall). In the interest of brevity, the main changes to the dataset are presented below and Figure 3 showcases the final form of the feature scheme:

i) Categorical features with a low expression that had been discarded in an earlier stage were reconsidered and binned together in batches with other low expression features whose target class representation is proportionally similar.

ii) Dimensionality reduction between correlated features, such as *Family*, *Single,* or *Spouse*. In that case, the most powerful feature was kept (*Spouse* in the previous example).

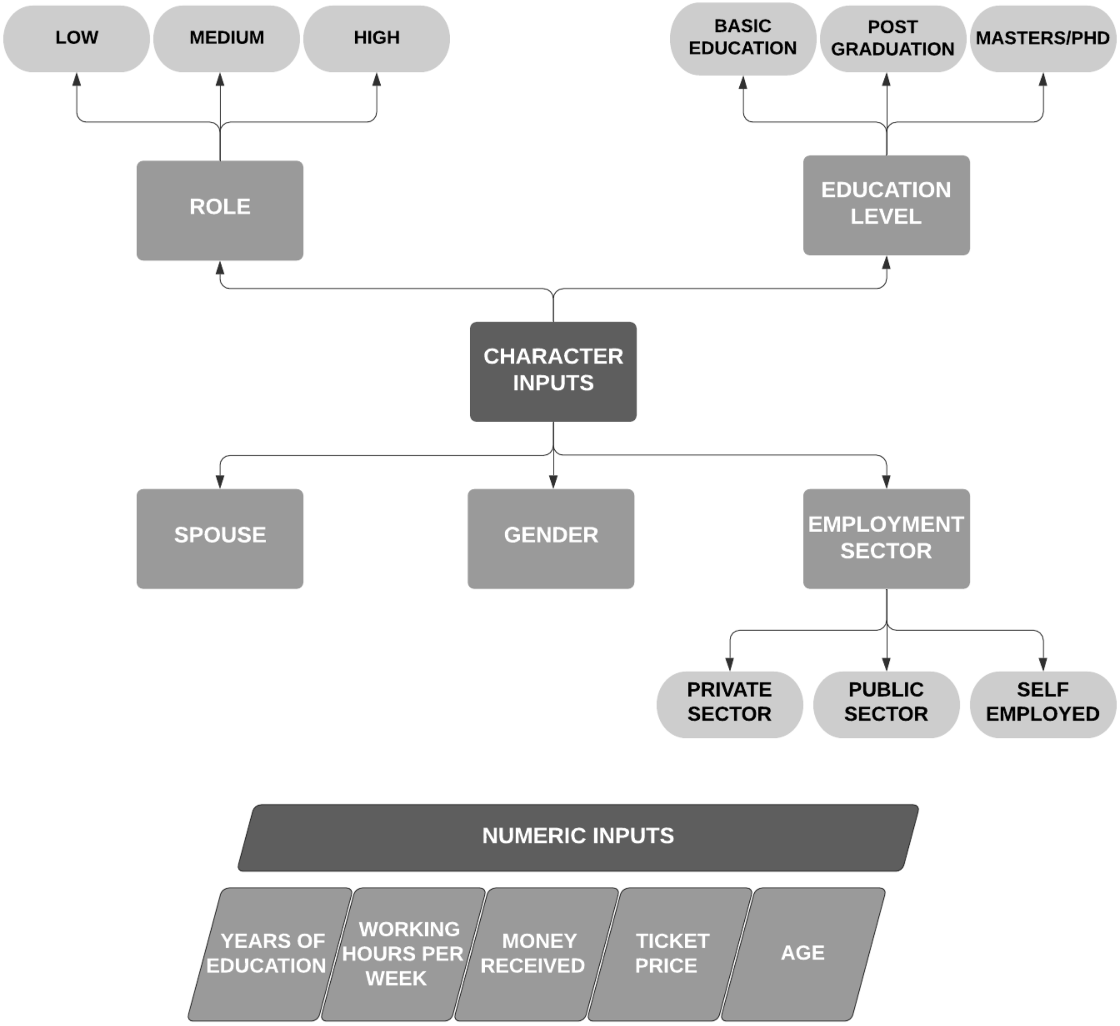

Figure 3: Final form of features on the dataset. Numeric features are *Years of Education*, *Working Hours per Week*, *Money Received*, *Ticket Price* and *Age.* Considered categorical features are *Spouse*, *Gender* and some subsets of previous variables *Role*, *Education Level* and Employment Sector.

III.6. Deployment

At the final stage of the CRISP-DM framework the trained model was given the testing set provided by the Newland Government. The model's predictions were uploaded to Kaggle where the Newland government compared their internal records with the model's output and returned the F1-score (micro) of the model 30% of the 10,100 test observations.

IV. Results

IV.1. Evaluation of Standalone Models

The main results of the model selection step with standalone models are briefly summarized in Table I. With F1 score micro as a success metric, it is possible to say that the best performing predictors on the sampled test set were DT (0.861), NN (0.854), and LR (0.842). At a first glance, there is a significant difference between the Training Score and the Validation/ Testing Scores in KNN that does not appear to be present in any of the other tested algorithms, which suggests overfitting of the model. The behavior of the considered Naive Bayes classifiers is also interesting to note. First and foremost, they were the fastest algorithms to perform classification. Additionally, BNB, NB, and MNB are the top 3 algorithms with the lowest overall precisions respectively. On the other hand, NB and BNB have the highest Recall Values. DTs, NNs, LR, and SVMs have small, but consistent, differences in results throughout. The hierarchy of values, from highest to lowest, for these algorithms is: DTs have higher F1-Score and Precision than NNs (with recall difference being 0.4 in favor of the latter), which in turn have higher values (F1 scores, Precision and Recall) than the values in LR, which in turn has higher values than SVMs.

Table I: Assessment metrics (F1 Score Micro, precision score, recall score and time required for training) for train set, validation set and test set of standalone models: K-nearest neighbors (KNN), Logistic Regression (LR), Gaussian Naive Bayes (NB), Bernoulli Naive Bayes (BNB), Multinomial Naive Bayes (MNB), Neural Network (NN) Decision Tree (DT and Support Vector Machine (SVM). DTs have the highest values on the F1 score and precision. The top performer in Recall is NB.

Test	KNN	LR	NB	BNB	MNB	NN	DT	SVM
F1 score – Training data	0.949 +/- 0.001	0.837 +/- 0.001	0.756 +/- 0.004	0.721 +/- 0.002	0.821 +/- 0.001	0.851 +/- 0.005	0.871 +/- 0.002	0.827 +/- 0.001
F1 score – Validation data	0.831 +/- 0.005	0.837 +/- 0.005	0.756 +/- 0.007	0.721 +/- 0.007	0.821 +/- 0.004	0.847 +/- 0.007	0.85 +/- 0.006	0.826 +/- 0.004
F1 score – Test data	0.835	0.837	0.765	0.726	0.824	0.848	0.852	0.827
Precision Training Data	0.966 +/- 0.003	0.707 +/- 0.003	0.491 +/- 0.004	0.451 +/- 0.002	0.651 +/- 0.003	0.74 +/- 0.028	0.813 +/- 0.017	0.673 +/- 0.004
Precision - Validation data	0.673 +/- 0.015	0.706 +/- 0.016	0.492 +/- 0.011	0.451 +/- 0.007	0.651 +/- 0.012	0.73 +/- 0.035	0.757 +/- 0.023	0.671 +/- 0.012
Precision - Test data	0.688	0.707	0.503	0.456	0.662	0.734	0.769	0.675

Recall – Training Data	0.813 +/- 0.004	0.536 +/- 0.005	0.844 +/- 0.005	0.808 +/- 0.003	0.531 +/- 0.004	0.575 +/- 0.038	0.593 +/- 0.021	0.523 +/- 0.009
Recall - Validation data	0.56 +/- 0.02	0.536 +/- 0.024	0.844 +/- 0.005	0.808 +/- 0.003	0.531 +/- 0.004	0.575 +/- 0.038	0.551 +/- 0.026	0.521 +/- 0.02
Recall - Test data	0.56	0.531	0.852	0.814	0.528	0.562	0.536	0.522
Time (seconds)	0.131 +/- 0.012	0.028 +/- 0.002	0.003 +/- 0.0	0.004 +/- 0.0	0.003 +/- 0.0	1.03 +/- 0.274	0.015 +/- 0.001	1.88 +/- 0.042

Figure 4 plots provide additional insights on the base learner's performance. NNs have the highest AUC in both ROC and average precision (AP) in Precision-Recall. The contrast between the F1-score and these tests suggest that, even though DTs seem to perform better at making predictions, NNs and, according to the ROC AUC, potentially LR are better at distinguishing between classes.

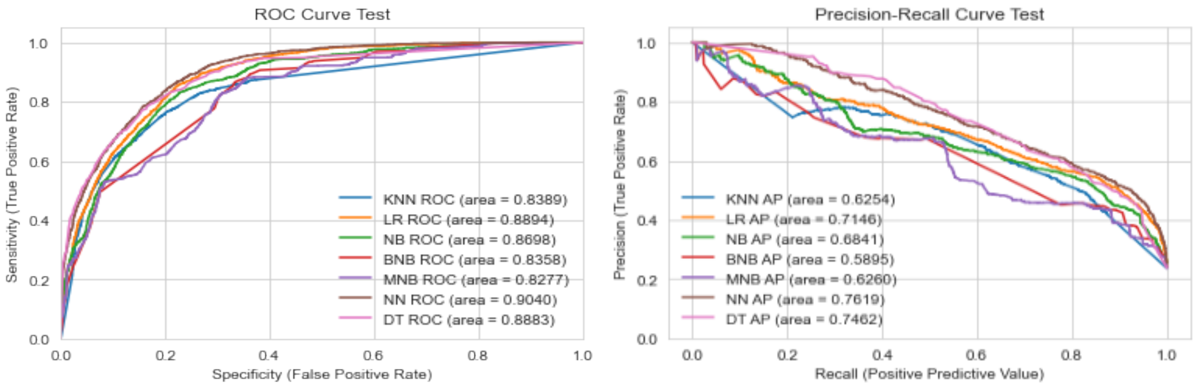

Figure 4: OC curve with respective ROC score (left) and Precision-Recall curve with respective AP score (right) for test set of evaluated standalone models.

IV.2. Ensemble methods

As there was no clear *best performer*, the authors opted to, on a first approach, test ensemble methods that have either DTs or NNs as base learners. performed the best in this classification problem with no clear *best performer*. Thus, the authors performed the same tests on Random Forests (RF), Adaptive Boosting (AdaBoost) and Gradient Boosting (that use DTs as their default single base learner) and Bagging NNs (a bagging ensemble algorithm using neural networks as its base learner). The results with these learners are summarized in Table II and their ROC and Precision-Recall curves are plotted in Figure 5.

Table II: Assessment metrics (F1 Score Micro, precision score, recall score and time required for training) for train set, validation set and test set of Bagging and Boosting ensembles: Bagging with Neural Networks as base learner (Bagg_NN), RF, AdaBoost and Gradient Boosting (GBoost).

Test	Bagg_NN	RF	AdaBoost	GBoost
F1 score – Training data	0.853 +/- 0.002	0.872 +/- 0.001	0.86 +/- 0.002	0.871 +/- 0.001
F1 score – Validation data	0.849 +/- 0.006	0.859 +/- 0.006	0.858 +/- 0.006	0.864 +/- 0.006
F1 score – Test data	0.845	0.858	0.856	0.865
Precision Training Data	0.744 +/- 0.013	0.826 +/- 0.005	0.771 +/- 0.004	0.797 +/- 0.004
Precision - Validation data	0.735 +/- 0.019	0.787 +/- 0.017	0.768 +/- 0.015	0.778 +/- 0.016
Precision - Test data	0.736	0.795	0.767	0.785
Recall – Training Data	0.579 +/- 0.017	0.583 +/- 0.007	0.58 +/- 0.008	0.611 +/- 0.005
Recall - Validation data	0.572 +/- 0.027	0.555 +/- 0.022	0.578 +/- 0.023	0.596 +/- 0.025
Recall - Test data	0.541	0.538	0.566	0.595
Time (seconds)	5.55 +/- 0.583	1.775 +/- 0.037	0.346 +/- 0.005	0.863 +/- 0.022

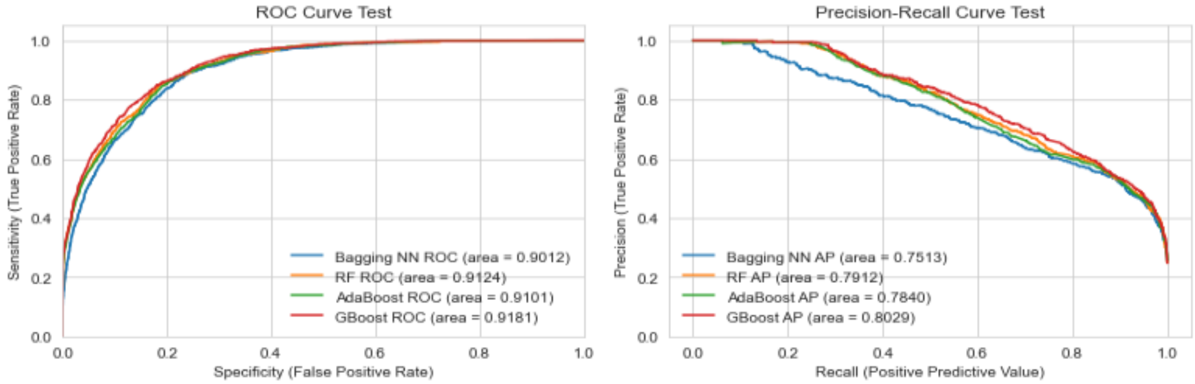

Figure 5: ROC curve with respective ROC score (left) and Precision-Recall curve with respective AP score (right) for test set of evaluated Bagging and Boosting Ensembles.

Unlike the previous set of results, the ensembles revealed GBoost as a clear winner that outperformed all other previously considered methods (base learners or not): GBoost Boosting had the highest F1 Score, the AUC in the ROC test and the highest AP on the Precision-Recall test. In fact, GBoost is only outshined by the stacking methods highlighted in Table III and whose ROC and Precision-Recall curves are plotted in Figure 6.

Table III: Assessment metrics (F1 Score Micro, precision score, recall score and time required for training) for train set, validation set and test set of stacked that performed better than GBoost: RF + NN, GBoost + SVM, GBoost + NN and GBoost + NB). GBoost + SVM performance is marginally better than GBoost with NB.

Test	RF + NN	GBoost + SVM	GBoost + NN	GBoost + NB
F1 score – Training data	0.87 +/- 0.0	0.869 +/- 0.0	0.847 +/- 0.01	0.869 +/- 0.0
F1 score – Validation data	0.858 +/- 0.0	0.863 +/- 0.0	0.845 +/- 0.0	0.862 +/- 0.0
F1 score – Test data	0.861	0.87	0.853	0.869
Precision - Validation data	0.764 +/- 0.02	0.774 +/- 0.02	0.732 +/- 0.03	0.77 +/- 0.01
Precision - Test data	0.783	0.794	0.808	0.788
Recall - Validation data	0.581 +/- 0.02	0.597 +/- 0.02	0.55 +/- 0.06	0.598 +/- 0.02
Recall - Test data	0.574	0.609	0.498	0.612
Time (seconds)	17.663 +/- 0.84	17.627 +/- 1	6.401 +/- 0.088	5.158 +/- 0.04

With an F1-Score of 0.87, the stacked ensemble of Gboost with SVM was the best performing model, followed by a stacked ensemble of GBoost with NB as a close second (F1-Score 0.869). The difference in performance in the different metrics is within the distance of one standard deviation, which suggests that both models are almost interchangeable. The degree of similarity between model performance is also observable in the ROC Curve and Precision-Recall curves observable in Figure 6.

Considering the similarity of GBoost with SVM and GBoost with NB, the authors opted to use the negligibly less powerful, but much faster, stacked Gboost with NB.

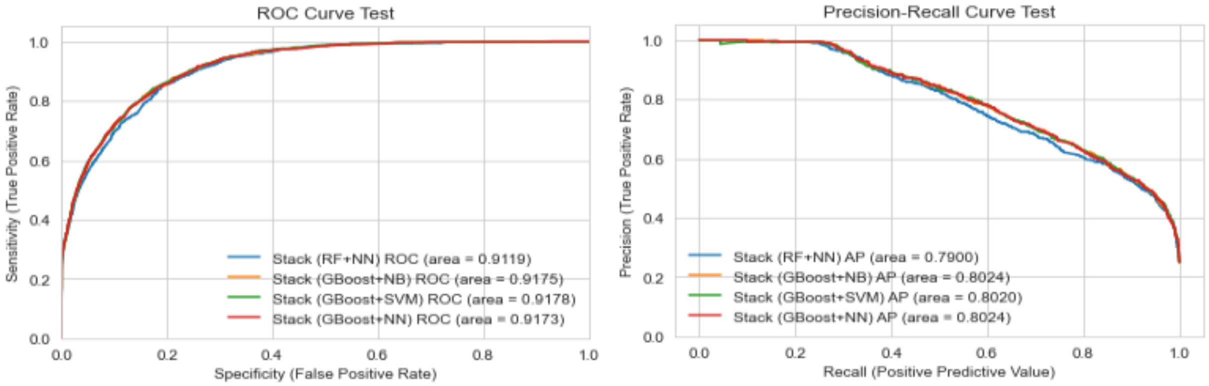

Figure 6: ROC curve (left) and Precision-Recall curve (right) of the stacked ensembles that had better F1-Score than Gboost.

IV.3. Evaluation of Deployed Model

Table IV summarizes the performance parameters of the optimized model. When compared with preoptimization results, it is noteworthy that there is an increase of Recall at the expense of a small decrease in the F1-score and Precision. Figure 8 shows that both the AUC ROC curve and the Precision-Recall AP values have increased to 0.919 (from 0.9173) and 0.8084 (from 0.8024) respectively.

Table IV: Assessment metrics (F1 Score Micro, precision score, recall score and time required for training) for train set, validation set and test set of the final model, after optimization.

Test	GBoost + NB Optimized Classifier
F1 score – Training data	0.877 +/- 0.001
F1 score – Validation data	0.869 +/- 0.005
F1 score – Test data	0.866
Precision - Validation data	0.775 +/- 0.013
Precision - Test data	0.773
Recall - Validation data	0.632 +/- 0.022
Recall - Test data	0.614
Time (seconds)	5.266 +/- 0.86

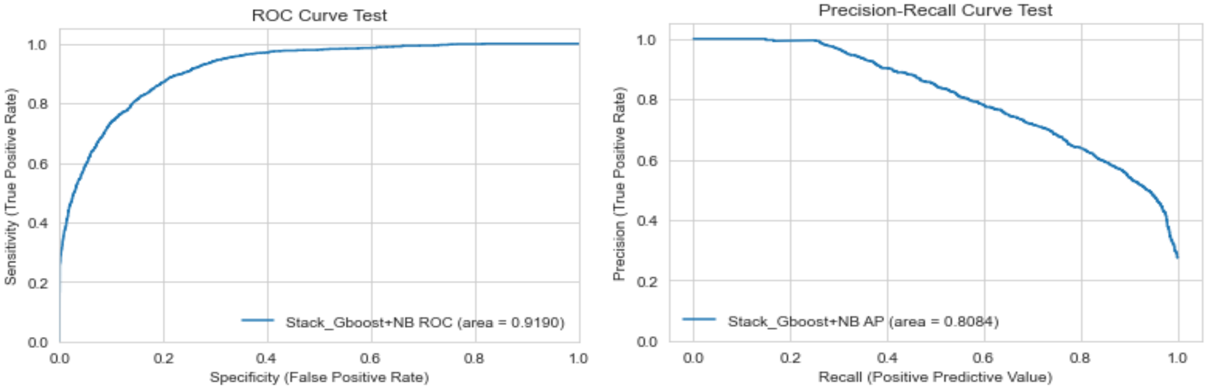

Figure 7: ROC (left) and Precision-Recall Curves of the Optimized stacked Ensemble of GBoost and Naive Bayes. The optimized model outperforms the non optimized model on these metrics.

The model's score on Kaggle is 0.86501, which is consistent with the F1 Score obtained from the split test dataset. Indeed, Figure 8 shows the misclassification rate for the training, validation and testing sets, whose difference has proven to be extremely slim, which further ensured that the model was not overfitting to the training data.

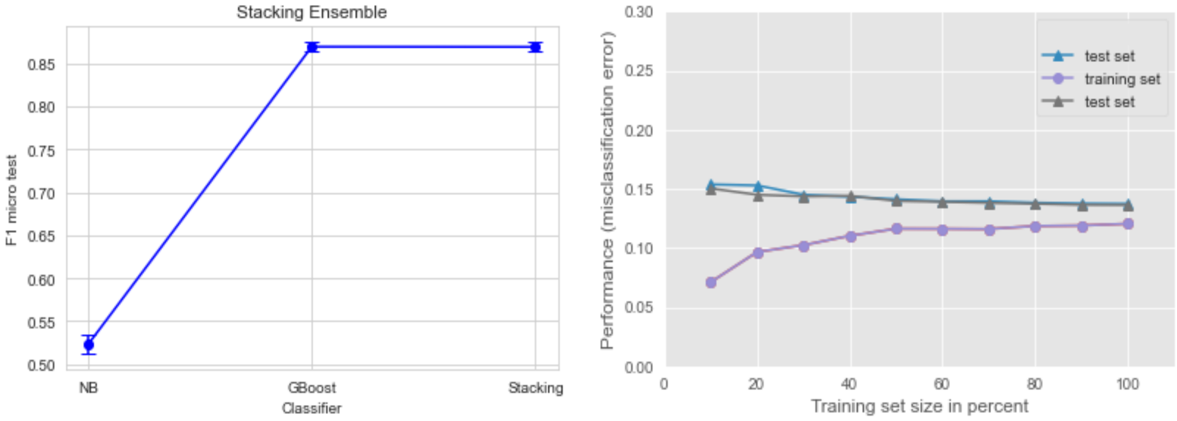

Figure 8: (Left) Plot showing F1 score achieved by each one of the base classifiers from the final stacking ensemble, as well as the F1 score achieved by the final model. Learning Curve (Right) allows us to see no sign of overfitting of model to the training data (please pay attention to the y-scale).

V. Discussion

In this section, the authors will look to provide a brief justification of some of the decisions made throughout the project. In particular, the section will focus: i) on the model selection process (i.e. why they were considered and why did they perform the way they did, ii) the properties of stacked ensembles, particularly the stack of GBoost and NB that justify the performance of the predictive model.

V.1. Selection and performance of base learners

The reasoning behind the choice of models heavily considered the fact that, even though income prediction tends to be addressed as a regression problem in the literature (Kibekbaev and Duman, 2016), the authors lacked the real values of income that would allow the adoption of a consistent regression framework and considered it as more resemblant of a classification problem (S. B. Kotsiantis et al., 2006) and the chosen algorithms reflect that:

i) KNN looks at the class of the n closest neighbors and assigns the majority vote. Even though KNN is widely used in classification problems, it is sensible to mixed data and high dimensionality (which would provide a possible explanation for the overfitting observed in the training data).

ii)LR calculates the probability of a sample having a positive or negative outcome if there if the data has a linear decision boundary. It is used often in classification problems and generally yields good results. Due to its high requirements, it is usually surpassed by other algorithms, which is reflected in the results (Feng et al., 2014).

iii) The logic behind Bayesian classifications was briefly covered in the Background section. In brief, the Bayesian classifier is the optimal classifier if a set of conditions are met. In reality, those conditions are relaxed to a significant degree. One of those relaxations is that the data that follows an unknown statistical distribution with unknown parameters can be fitted to a known distribution. To try Bayesian classifications, the authors chose the Bernoulli, Multinomial and Gaussian distributions. Repeated instances of the Bernoulli distribution form a Binomial distribution which is a Multinomial distribution where there are 2 possible outcomes (0 and 1, same as the dataset provided by the Newland Government). The Gaussian classifier is also considered because Normal approximation is possible from Binomial distributions. These classifiers had the highest values in Recall (Percentage of true 1s that were labeled correctly) but Precision (percentage of values labeled as 1 that are truly 1) lower than 0,50 in BNB and NB. These results indicate that the probability threshold that maximizes the accuracy of the model leads to the generation of a high number of false positives and a low number of false negatives, which is equivalent to having low AP score (ranging from 0.59 in BNB to 0.68 in MNB. Rather than being good at distinguishing between classes, the Bayesian classifiers seem to be biased towards predicting positives. This result showcases the issue that a skewed class distribution on the training dataset can have on the overall performance of the algorithms. Indeed, most classification algorithms are design to optimize overall accuracy, often causing a lower prediction performance on the minority class, which is represented by a high false-positive rate (Lee et al., 2009).

iv) SVM is a non-probabilistic model that is optimized by a convex loss function that has applications in various fields of expertise. Due to the model's properties, there does not seem to be any obvious or blatant justification against the use of SVMs (Bennett and Campbell, 2000). The algorithm's performance was also on par with well-performing models but the algorithm had lower precision which translates into more difficulty in predicting the minority class (which, in the case of the Newland government, represents future payers of higher income tax bracket).

v)NN and DT are more complex models that performed the best in all metrics on a base level (F1-score, ROC AUC and Precision-Recall AP). As both NNs and DTs are models with large potential for optimization, authors relied on these models to create ensembles that could potentially increase the model's predictive power.

V.2. Stacked ensembles and GBoost with NB

Results obtained on the model selection phases pinpointed GBoosted Decision Trees (as base learners) as the best predictors of the classification problem. GBoost is a learning procedure that consecutively fits new models to provide a more accurate estimate of the response variable. The main idea behind it is to build the new base-learners to be maximally correlated with the negative gradient of the algorithm's loss function (Natekin and Knoll, 2013). The author's efforts led to the use of GBoost within stacked ensembles with different base learners.

Stacked ensembles allow to average out noise from multiple models, thus improving the generalizable signal. These algorithms use the predictions of different learning algorithms as inputs to a second-level learning algorithm. Consequently, this second-level algorithm combines the model predictions (from previous algorithm(s)) to generate a final set of predictions. The traditional rationale of stacking algorithms is to combine *weak learners* (as base learners) as a way of improving their performance individually. However, recent studies have shown promising results by combining strong yet diverse models (Gunes et al., 2017). This approach resulted in improvements in models' prediction accuracy, beyond the capabilities of any single base learner. With the advent of these models, the mindset of a classification or regression problem shifted from finding the single model with the best prediction accuracy, to finding a collection of several complementary models that work well together.

The higher predictive performance achieved by stacking ensembles comes at the cost of very high training times. The high training times were noticeable as training times increased considerably in comparison to the time required to train e.g. standalone GBoost. It is advised to consider the tradeoff between very slight increases in the predictive performance of the stacking ensemble and time and the computational expense required to train it.

The results revealed slight improvements in predictive power when Gboost was combined with NB and SVMs. Although results revealed a slight advantage to the stacked ensemble of Gboost with SVMs, the differences between the use of SVMs and NBs were almost negligible, which led to the decision to proceed to model optimization with a stacked ensemble of GBoost and NB due to the much lighter computational burden.

The performance of the stacked ensemble of GBoost with NB was surprising. Upon additional research, it noteworthy that work performed by other authors with a dataset that has similar characteristics to the one compiled by the Newland Government - a binary classification problem with a predominance of categorical variables and large size - had a boosting algorithm with Naive Bayes as the model with the lowest error rate (Kim and Kim, 2004). Other authors have noted that the key may be in the stability of the algorithm, at least in some classification problems. NB is a relatively stable classifier even when in the presence of small changes to the training data, which is the opposite of what happens with a decision tree. Therefore, it has been hypothesized that the addition of tree structures to a stable classifier makes way to the reduction of bias and increase variance (Ting and Zheng, 2003).

VI. Conclusion

The use of stacking ensembles of carefully chosen strong and diverse models has been recently widely used, as they have proven to boost the predictive performance of multiple machine learning algorithms and produce highly accurate predictions. Here, a Gradient boosting model (using decision trees as its base estimator) was combined with a Naive Bayes Classifier in a stacking ensemble using Logistic Regression as a meta-classifier, in order to create the final classification algorithm. The authors tried to maximize the number of correct predictions, despite the computational burden that could bring, as it was considered that having the best performing model is important for an accurate prediction of the taz returns of the citizens. Despite being able to achieve a good result, the authors are confident that more work could be put towards optimizing the model.

In future work, authors could attempt to combine multiple models in an initial stage of the stacking ensemble. Each of these models could make use of a training set whose feature selection had been optimized for the model in question, and whose hyperparameters had been optimized to increase the performance of that model alone. However, variation is highly appreciated in the initial models of the stacking ensembles as this is what allows the ensemble to attribute different weights to different models that are best optimized to predict the class of a certain set of instances. Thus, even if the initial stage of these stacking ensembles makes use of the same model, it could be beneficial that those are trained with different hyperparameters and/or training sets with different feature subsets. This can be achieved with bootstrapping of the training instances from the training dataset and/or cross-validation. Further stage(s) would take the predictions of the previous layer to train models that would, ultimately, output their predictions to a final model. This is a process that takes a lot of trial and error.

Given the nature of our dataset, it is suggested that dividing it in 3 sets would be beneficial, as there is a group of people who were offered money to enter the program, a set of people who paid money to enter the program and a set of people who neither paid nor were paid to enter the program. The authors hypothesize that each of these sets of people present different characteristsic that make them differently valuable for the program, and having them all in the same subset of data could contribute to a worse performing model, whose objective is to find the features that are the best predictors for a person whose income is above or below average. Thus, as further step to tackle this classification problem, this would be an approach it should be tried next to try getting a model with better performance. Possibly, a stacking algorithm, with 3 models, each optimized to one of the three datasets with a different set of features, could work well.

VII. References

Alstadsæter, A., Johannesen, N., Zucman, G., 2019. Tax Evasion and Inequality. Am. Econ. Rev. 109, 2073–2103. https://doi.org/10.1257/aer.20172043

Bennett, K.P., Campbell, C., 2000. Support vector machines: hype or hallelujah? ACM SIGKDD Explor. Newsl. 2, 1–13. https://doi.org/10.1145/380995.380999

Chakrabarty, N., Biswas, S., 2018. A Statistical Approach to Adult Census Income Level Prediction, in: 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). Presented at the 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), IEEE, Greater Noida (UP), India, pp. 207–212. https://doi.org/10.1109/ICACCCN.2018.8748528

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R., 2000. Step-by-step data mining guide 76.

Domingos, P., Pazzani, M., 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss 28.

Drummond, C., Holte, R.C., 2004. What ROC Curves Can't Do (and Cost Curves Can). ROCAI 19–26.

Fan, J., Upadhye, S., Worster, A., 2006. Understanding receiver operating characteristic (ROC) curves. CJEM 8, 19–20. https://doi.org/10.1017/S1481803500013336

Feng, D., Gourieroux, C., Jasiak, J., 2008. The ordered qualitative model for credit rating transitions. J. Empir. Finance 15, 111–130. https://doi.org/10.1016/j.jempfin.2006.12.003

Feng, J., Xu, H., Mannor, S., Yan, S., 2014. Robust Logistic Regression and Classification. Robust Logist. Regres. Classif. Adv. Neural Inf. Process. Syst. 27, 9.

Ghahramani, Z., 2015. Probabilistic machine learning and artificial intelligence. Nature 521, 452–459. https://doi.org/10.1038/nature14541

Guillaud, E., Olckers, M., Zemmour, M., 2020. Four Levers of Redistribution: The Impact of Tax and Transfer Systems on Inequality Reduction. Rev. Income Wealth 66, 444–466. https://doi.org/10.1111/roiw.12408

Gunes, F., Wolfinger, R., Tan, P.-Y.T., 2017. Stacked Ensemble Models for Improved Prediction Accuracy. Proc Static Anal Symp 19.

Haibo He, Garcia, E.A., 2009. Learning from Imbalanced Data. IEEE Trans. Knowl. Data Eng. 21, 1263–1284. https://doi.org/10.1109/TKDE.2008.239

Hu, L., 2018. Non-IID Recommender Systems: A Machine Learning Approach (University of Technology of Sidney, Faculty of Engineering and Information Technology). Sydney, Australia.

Jiawei Han, Jian Pei, Micheline Kamber, 2011. Data Mining: Concepts and Techniques, The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, Burlington, MA.

Jordan, M.I., Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. Science 349, 255–260. https://doi.org/10.1126/science.aaa8415

Kibekbaev, A., Duman, E., 2016. Benchmarking regression algorithms for income prediction modeling. Inf. Syst. 61, 40–52. https://doi.org/10.1016/j.is.2016.05.001

Kibriya, A.M., Frank, E., Pfahringer, B., Holmes, G., 2004. Multinomial Naive Bayes for Text Categorization Revisited, in: Webb, G.I., Yu, X. (Eds.), AI 2004: Advances in Artificial Intelligence, Lecture Notes in

Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 488–499. https://doi.org/10.1007/978-3-540-30549-1_43

Kim, H., Kim, J., 2004. Combining Active Learning and Boosting for Naïve Bayes Text Classifiers, in: Li, Q., Wang, G., Feng, L. (Eds.), Advances in Web-Age Information Management. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 519–527.

Kotsiantis, S., Kanellopoulos, D., Pintelas, P., 2006. Handling imbalanced datasets: A review 12.

Kotsiantis, S.B., Zaharakis, I.D., Pintelas, P.E., 2006. Machine learning: a review of classification and combining techniques. Artif. Intell. Rev. 26, 159–190. https://doi.org/10.1007/s10462-007-9052-3

Lee, M.S., Rhee, J.-K., Kim, B.-H., Zhang, B.-T., 2009. AESNB: Active Example Selection with Na&#x0EF;ve Bayes Classifier for Learning from Imbalanced Biomedical Data, in: 2009 Ninth IEEE International Conference on Bioinformatics and BioEngineering. Presented at the 2009 Ninth IEEE International Conference on Bioinformatics and BioEngineering (BIBE), IEEE, Taichung, Taiwan, pp. 15–21. https://doi.org/10.1109/BIBE.2009.63

Natekin, A., Knoll, A., 2013. Gradient boosting machines, a tutorial. Front. Neurorobotics 7, 21. https://doi.org/10.3389/fnbot.2013.00021

Nouretdinov, I., Vovk, V., Vyugin, M., Gammerman, A., 2001. Pattern Recognition and Density Estimation under the General i.i.d. Assumption, in: Helmbold, D., Williamson, B. (Eds.), Computational Learning Theory. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 337–353.

Provost, F., Fawcett, T., Kohavi, R., 1998. The Case Against Accuracy Estimation for Comparing Induction Algorithms. Proc. Fifteenth Int. Conf. Mach. Learn. 9.

Ting, K.M., Zheng, Z., 2003. A Study of AdaBoost with Naive Bayesian Classifiers: Weakness and Improvement. Comput. Intell. 19, 186–200. https://doi.org/10.1111/1467-8640.00219