Predictive Report: Newland

Beatriz Chumbinho (r20170867@novaims.unl.pt), Inês Costa (r20170775 @novaims.unl.pt), Maria Leonor Morgado (r20170871@novaims.unl.pt), Rodrigo Matias(r20170880@novaims.unl.pt)

***Abstract*— We were proposed to predict whether a Newland resident has an Income above or below the average of all the residents so that a binary tax rate can be applied to them by the government, according to their income class.
To conquer this objective, we started by applying some changes to the dataset given to us. Firstly, data cleaning which includes missing values imputation and outlier removal then, data preprocessing, having done feature engineering, feature encoding and feature selection. When choosing the features that influenced the population's income through various feature selection methods, we realized that these were related to age, education, the Newland experience and current life status, like marital status and profession.
Finally, we concluded that the classifier that reproduced the best prediction, was the Gradient Boosting algorithm which led to an 0.8630 weighted f1-score in the final prediction submitted.**

Keywords : Machine Learning, Decision Tree, Predictive Models, Gradient Boosting, Income, Classifiers

I. Introduction

The world is going through a major change, a new planet similar to Earth has been found and is now, in 2048, being populated. This planet, Newland, needs to be developed in order to be a new home, so a lot of action is necessary, namely, Newland has to be financially sustainable. To accomplish this, the government decided that residents had to start paying taxes according to their income, 15% of the income if this is below average and 30% otherwise. But, to make this system work they need to know in which class each resident is, and that is where we engage, by creating a model that predicts the class each resident belongs to.

II. Background

The data has to go through pre-processing before any classifier is applied, in order to get the best predictive model. After dealing with the missing values and making some feature engineering, we used *One Hot Encoding* on our categorical variables so that we can use them to predict the 'Income' class. These variables have no ordinal relationship so, this encoding represents the categorical variables as binary vectors, but first each unique category in each variable has to have assigned an integer value. To remove outliers in metric features, the *Local Outlier Factor* (*LOF*) was applied. This algorithm calculates the local density of a point and compares it with its neighbors. If a point has quite lower density that its neighbor is considered an outlier.

III. Methodology

Note: All the study was conducted using methods from scikit-learn package from python .

In order to predict the variable *'Income',* our team was given a dataset with 22400 rows and 14 variables. To better understand the context, in the fig.1 at the end of this page it is possible to have an overall view on the dataset.

a) ***Splitting the data:*** all rows and columns but the target were assigned to an 'X' variable, and all the rows of the target column were assigned to an 'y' variable. The training dataset was splitted into a 'X_train', 'y_train' which contains 80% of the data and the 'X_test' ,'y_test' having the remaining 20%. To split the data, the train_test_split was used with the X and y variable used as parameters to the train and test, the variable 'y' as value for the *stratify* parameter, a *random_state*=15 and a *test_size* of 0.2 in order to keep 80% for the training.

	CITIZEN_ID	Name	Birthday	Native Continent	Marital Status	Lives with	Base Area	Education Level	Years of Education	Employment Sector	Role	Working Hours per week	Money Received	Ticket Price
0	12486	Mr. Adam Glover	July 1,2003	Europe	Married	Wife	Northbury	High School + PostGraduation	13	Private Sector - Services	Repair & constructions	40	0	2273
1	12487	Mr. Cameron McDonald	January 25,2006	Europe	Married	Wife	Northbury	Professional School	12	Public Sector - Others	Repair & constructions	40	0	0
2	12488	Mr. Keith Davidson	May 10,2009	Europe	Married	Wife	Northbury	Professional School	12	Private Sector - Services	Sales	46	0	2321

fig.1

b) ***Missing values*** - After analysing the dataset we noticed that *'Base Area'*, *'Role'* and *'Employment Sector'* had missing values - we needed to convert some characters as '?' to nan - , since those are categorical variables we imputed them with the mode.

c) ***Unbalanced learning:*** in order to keep aware about the purpose of the study, checking the unbalanced learning was important to understand what was the most common classification of the target attribute - 0 or 1. There were more 0 than 1, this is, more people with an income below the average than people with an income above the average.

```
1  df['Income'].value_counts()

0    17089
1     5311
```

fig.2

d) **Outliers** - The first outlier removal method tried in our study was the *visualization method* (through plots) however, the percentage removed after completing this step was 1.4% which our team considered wispy. Then, the *LOF method* was applied to the data, starting with a contamination of 0.016 - in order to remove at total 3% of the data- and the both methods were combined. However, our team considered the visualization method a little ambiguous since some attributes were very hard to fully visualize, as it is possible to confirm in the fig.3. This way, our last decision was to use only the *LOF* method, with a contamination of 0.03 which covers our goal to discard ~ 3% of the data. The percentage of data kept after removing outliers was 0.97.

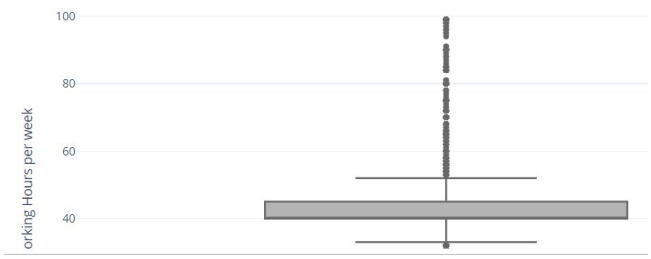

fig.3

e) **Checking inconsistencies** - We verified that there were no duplicate or incoherent rows in the dataset provided. We looked for incoherences in the values of *'Age'* comparing to the *'Years of Education'*, *'Marital Status'* and *'Lives With'* , ensured that people with a career in *'Army'* work for the *'Public Sector'*, and also we guaranteed that people with *'Money Received'* greater than 0 needed 'Ticket Price' to be equal to 0, or vice-versa. There were not found incoherences through the data during the analysis.

f) **Checking Correlations:** We checked the correlation between our variables and the target variable. Using Spearman correlation for metric features and plotting the data for categorical ones.

g) **Feature Engineering:** in this section some new variables were created and another were modified. *'Age'* were created through the *'Birthday'* attribute. *'Gender'*, was engineered to represent a 0 or 1 depending if the observation corresponds to a woman or a man. The *'Employment Sector'* variable was adapted and now it represents if the person works for the Public/private, if he/she is self-employed or unemployed. Since on this study it is easy to divide people in groups based on 'Money Received' and *'Ticket Price'* , three new variables were created - *'Group A'*, *'Group B'* and *'Group C '* - corresponding, respectively if the person was a volunteer, if the person was paid or if the person was paying a ticket - which we ended up not using it in the final model. Finally, *'money_dif'* is a created feature that aims to represent the difference

between the *'Money Received'* and the *'Ticket Price'* of each observation.

h) ***Deleted variables*:** Some variables were deleted during this step - *'Name'* (creation of *'Gender'*), *'Birthday'* (creation of *'Age'*), *'Ticket Price'* , *'Money Received'* the last two were removed due to their high correlation with *'money_dif'* and *'Base Area'* was also deleted by consequence of its high cardinality.

i) **Encoding** - The result of encoding the categorical variables with *One Hot Encoding* was the creation of a column for each different category in every categorical variable: *['Native Continent', 'Marital Status', 'Lives with', 'Employment Sector', 'Role'*]. These new columns are binary, having the values 0 or 1 depending on its category in that variable. At the end of this step, these initial categorical features are removed from the datasets because they are already represented with the new created columns. Below it is possible to observe a sample in fig.4 that shows how the features look like after encoding.

	x0_Europe/Asia	x0_Other	x1_Alone	x1_Children	x1_Other	x1_Wife/Husband	x2_Army	x2_Primary
13190	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
7657	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
5918	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
20123	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
11008	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...

fig.4

j) **Normalization** - We decided to use the *StandardScaler*, which standardizes features by removing the mean and scaling to unit variance, in order to remove the differences between the features. So, the same dataset was obtained but with variables within the same scale of value - from 0 to 1. Past this, we ended up by choosing the *Gradient Boosting Algorithm* which performs worse with normalized data.

k) ***Feature selection*** - This is a crucial step to get the best predictive model and so we decided to apply to our preprocessed dataset a variety of different methods to guide our choice. After checking the correlations and using *RFE, Lasso, Ridge* feature selection methods we ended up by choosing this columns: *'x1_Married'*, *'money_dif'*, *'Years of Education'*, *'Age'*, *'Working Hours per week'*, *'x4_Management'*, *'x4_Professor'*, *'x2_Husband'*, *'x4_Agriculture and Fishing'*, *'x4_Other services'*, *'Gender'*, *'x3_Public'*. To justify this choice, *'x4_Management'* and *'x4_Professor'* were both selected by the *RFE* and the *Ridge*. *'Age'*, *'Working Hours per week'* and *'money_dif'* were selected by *Lasso* and the remaining ones by *Ridge*. Only the *'x3_Public'*, *'Years of Education'* and *'x4_Other services'* did not appear in any of the methods but after a trial-error process we decided to include, since the accuracy of the model responded well to them.

Note: All the preprocessing steps mentioned above were performed for both X_train and X_test - our validation - dataset.

I) **Test dataset:** To provide the final prediction results in Kaggle, before applying the model to the original test dataset we had to preprocess the data. All the preprocessing performed in the original train dataset was also made in this dataset, except the outliers removal.

IV. Results

1) ***Classifier choice:*** Tested a great variety of classifiers, like *MLP, neural networks (NN), KNN, SVC, random forest (RF), decision tree (Dtree)*, and also some which combine more than one method, *ensemble methods* such as, *gradient boosting (GB), adaboost (AB), bagging.* In the fig.5 it is possible to verify the respective score of each classifier, which is the mean accuracy on the X_test dataset. Note: these models were fitted with no parameter changes.

Score	Model
0.870089	GB
0.862723	AB
0.852009	RF
0.843973	KNN
0.827232	Dtree
0.814063	NN
0.803348	SVC

fig.5

2) **Gradient Boosting (GB)**: produces a prediction model in the form of an ensemble, typically *decision trees*. It creates a single leaf and then *GB* builds a tree - usually 8 to 32 leaves - from the errors made on the previous tree. After that, the algorithm scales the trees. *GB* kept building trees until it achieved a threshold or failed to improve the fit.

3) ***The remaining models:***

   a) *Ada Boosting (AB):* creates stamps which have different importance in the final classification and the precious mistakes in other stamps are always taken into account.

   b) *Random Forest (RF)*: this model creates full size trees where each one is worth the same and the order in which trees are built is not important for the algorithm.

   c) *KNN Classifier*: the input consists of the *k* closest training examples in the feature space. The output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors.

   d) *Decision Trees (Dtrees):* is used to visually and explicitly represent decisions making.

   Developing a tree requires deciding on which features to select and what conditions to use for splitting, along with knowing when to stop.

   e) *Multilayer Perceptron (NN):* A multilayer perceptron it's a neural network with more than 1 layer. The network learns through exposure to various situations by adjusting the weight of the connections between the neurons grouped into layers.

   f) *Support Vector Classification (SVC):* The purpose of this algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The main goal is to find a plane that has the maximum distance between data points of both classes.

Even after the parameter tuning in these previous classifiers we were never able to achieve an equal accuracy of the *Gradient Boosting*.

4) ***Feature Importance:*** Taking into account the models tested above, we created a board showing their respective features importances and after that, we tested the different models combining the different features. The figure below it is possible to understand that our previous choice of features match the needs of the *Gradient Boosting*, our selected model:

feature	importance_GB	importance_AB	importance_RF	importance_Dtree
x1_Married	0.368	0.02	0.072	0.204
money_dif	0.275	0.28	0.145	0.169
Years of Education	0.193	0.18	0.132	0.138
Age	0.062	0.12	0.248	0.181
Working Hours per week	0.036	0.06	0.120	0.110
x4_Management	0.018	0.02	0.021	0.011
x4_Professor	0.013	0.02	0.019	0.010
x2_Husband	0.006	0.04	0.011	0.006
x4_Agriculture and Fishing	0.006	0.02	0.005	0.005
x4_Other services	0.005	0.02	0.008	0.006
x4_IT	0.003	0.02	0.006	0.006
Gender	0.003	0.04	0.012	0.011
x3_Self-Employed	0.002	0.00	0.009	0.015
x3_Public	0.002	0.00	0.007	0.007
x0_Europe	0.001	0.00	0.008	0.011
x3_Private	0.001	0.00	0.010	0.016
x4_Cleaners & Handlers	0.001	0.02	0.003	0.004
x4_Security	0.001	0.02	0.004	0.005
x2_Children	0.001	0.02	0.009	0.002

fig.6

5) ***Gradient Boosting Parameters****:* after some trial and error we ended up by choosing a *learning rate* of 0.25 and 99 *estimators.* Which led to a 0.8742 mean accuracy in X_train dataset and 0.8763 on the X_test dataset. Meaning that by fine tuning the parameters we were able to improve the mean accuracy in the X_test by 0.006211.

6) ***The effect of normalization on Gradient Boosting:***

We tested the use of Normalization on Gradient Boosting and the mean accuracy scores on the X_test dataset we obtained were: Using *MinMax Scaler* 0.82745, with a *Standard Scaler* we got 0.85067 so this means that our algorithm was losing 0.02563 point in mean accuracy using this scaler. This is a common behavior in Decision Trees family algorithms.

7) ***Final Results of our model:***

TRAIN

	precision	recall	f1-score	support
0	0.89	0.95	0.92	13671
1	0.80	0.63	0.70	4249
accuracy			0.87	17920
macro avg	0.84	0.79	0.81	17920
weighted avg	0.87	0.87	0.87	17920

```
[[12994   677]
 [ 1577  2672]]
```

VALIDATION

	precision	recall	f1-score	support
0	0.89	0.95	0.92	3418
1	0.81	0.63	0.71	1062
accuracy			0.88	4480
macro avg	0.85	0.79	0.81	4480
weighted avg	0.87	0.88	0.87	4480

```
[[3257  161]
 [ 393  669]]
```

fig.7

a) *Train*: Our reference is the *f1- score*, which has a value of 0.87 for the X_train, being better at predicting targets with a value of 0 (0.92) than targets classified as 1 (0.70). Regarding the ***confusion matrix***, it is possible to confirm that in 13671 targets with the value 0, we predicted well 12994 (95%) and failed 677 (5%). For the target variables classified with the value 1, we predicted well 2672 (60%) out of 4249 and failed in predicting 1577 (40%).

b) *Validation:* We achieved a value of 0.88 for the f1-score in the X_test, which is also better at predicting targets with a value of 0 (0.92) than targets classified as 1 (0.71). Observing the ***confusion matrix***, we can affirm that in 3418 targets with the value 0, we predicted well 3257 (0.95%) and failed 161 (5%). For the target variables classified with the value 1, we predicted well 669 (63%) out of 1062 and failed in predicting 393 (37%).

V. DISCUSSION

We are now **able to understand which are the variables that better help predicting if the income of a person is below or above the average.** Those variables are, specifically, age, whether that person paid or not to enter in the Newland experience, the education level,the marital status, the time spent at work and certain professions also determined the probability of someone receiving more than the average. People who are married tend to have a higher probability of having an income above the average, as well as the people with higher levels of education. Regarding the professions, some of them such as management positions or professors have a higher probability to receive more, on the other hand people who work as Cleaners, Handlers, Securities have an higher probability of receiving below average. Naturally, younger people are much more frequent to be classified with a 0 relating to their income, so they tend to receive below the average. People who spent low time working showed evidence of a bigger probability to receive less than the ones who work during more hours per week.

Based on the categorical plots that we did in the exploration steps of the dataset we are able to say that people who paid nothing to enter Newland but also have received nothing tend to have an income classified as 0, so only 20% of people within this group have an income above the average. People who paid nothing to enter Newland but were paid to go are the ones with an higher probability of having an income classified as 1, so to have an income above the average, approximately 60% of people within this group receive above average. The ones who have not received to enter Newland but have paid have, approximately, a probability of 50% to belong to the category of 0 or 1.

There was much more data from people receiving below average than above which led to some deficiencies in our model, being much better at predicting if someone receives below than above average. Since our main focus was to improve the weighted f1-score we did not tackle this problem of **unbalanced learning** as much as we should have in a real-world scenario.

VI. CONCLUSION

With our study results it is **possible to predict if someone has an income above or below the average**. This way, the **government has this to support their decision** regarding who

pays 15% of taxes and who pays 30% of taxes in the Newland planet.

We propose that new researches direct their efforts into a better prediction of above average incomes, approaching with more precision the targets -incomes- classified as above average.

After all in our last prediction (test dataset provided) we got a 0.86303 *f1 score* which we were well pleased with.

VII. REFERENCES

[1] Alina Lazar. Income Prediction via Support Vector Machine, Youngstown State University

[2] scikit-learn, Machine Learning in Python

[3] Roberto Henriques. Ensemble Classifiers

[4] Carina Albuquerque. Feature selection, notebook

[5] Carina Albuquerque. Model assessment and metrics, notebook