Newland residents' income predictions

Catarina Natário Moreira (m20201034@novaims.unl.pt), João Paulo Guerreiro Aredes César(m20200669@novaims.unl.pt), Maikel Sousa (m20200735@novaims.unl.pt)

Abstract— In the last few years, machine learning has become one of the most important and promising developments to emerge in the field of artificial intelligence. It's not surprising that its applications are becoming more widespread day by day in every business sector, always with new and more powerful tools and results. In this machine learning paper, we analyzed the real income (whether it is above or below the average) of each resident of the Newland, so that we could create a predicting model to apply to people on their way to Newland. This model aims to find out if the income of each citizen is above average or not. In this way, a binary tax rate may subsequently be applied depending on the inhabitant's income. For the construction of this model, we used variables that had a greater meaning for the model to be realistic. We used Years of Education, Money Received, Employment sector, Education Level, among others features.
We started by applying data cleaning and data preparation knowledge explored in Data Mining classes. Posteriorly, we used and compared regression methods such as Logistic Regression, k-Nearest Neighbours (kNN), Decision Trees and Random Forest Regression, among others. We predicted the income using stacking algorithm (with F1-score average with better results). We will present the details of the prediction questions, the analysis, testing and validation results for the different algorithms in this paper. In addition, we will also discuss the significances of our approach and methodology.

Keywords : Decision Tree, Machine Learning, Model, Predictive Model, Classification Methods

I. Introduction

Over the years, planet Earth has undergone some climate changes and has become unfeasible for all forms of life. Thus, later on, a new planet with conditions similar to planet earth was discovered. In this way, a mission was developed to inhabit the new planet. Years later, the Newland government decided to make the new city more financially sustainable and thus applied a binary tax rate to all its residents and future residents. This tax had a percentage that varied depending on the income of each citizen. It was, therefore, that we developed a model that could predict whether the income of habitant of the new city was higher than average or not.

Although machine learning has become a scientific discipline, the effective communication of its ideas remains an art.
In this work, we review work in machine learning on methods for handling data sets containing large amounts of irrelevant information. We focus on two key issues: the problem of selecting relevant features, and the problem of selecting the best and more relevant models.

The dataset we had available and in which we worked to test and develop the models has approximately 10100 examples and fifteen features/attributes such as Age, Education Level advanced, Ticket Price, Money Received, etc. These variables do not all have the same level of importance. In this way, we had to check those that were most relevant so that we could obtain a reliable model with good results.

Therefore, this rich and didactic dataset should be sufficient to establish a classification model to accurately predict the income of each citizen who is on the way to the new city.

In this machine learning paper, we predicted whether or not the citizen's income is above average using regression methods such as Logistic Regression, k-Nearest Neighbours (kNN), Decision tree and Random Forest, among others. We predicted the income using the stacking algorithm. We will present the details of the analysis, and the testing and validation results for the different algorithms below. We will also describe the results that have been made and present a general table that we use to compare the different models. In addition, we will also discuss the significances of our approach and methodology.
So, What is the best way and the best model to predict if the income is or not above the average?
This is the question we intend to respond. To do this, we will test several models and different strategies of variables to obtain the best possible performance and make the new city more financially sustainable.

II. Background

To carry out our work, we used models taught only in class. In addition to the jupyter notebooks provided by the professors at the practical classes, we also used the stackoverflow as a learning tool. Finally, explored the documentation on each model, changing the various parameters related to it.

III. Methodology

To carry out this project, we follow a set of steps that belong to the Machine Learning Process: Data Access, Exploration and Understanding, Data preparation and Modelling.

A. Data pre-processing

We started by exploring the dataset that was provided to us and the first action we took was to create a new 'Age' column that tell us the age of the citizen in the year 2048. Subsequently, we added this new column to the database.

In general, the data provided to us and which will be analyzed may have errors both in the original record or initial processing. Unfortunately, after analyzing the presented dataset, we discovered that we had three columns, base area, employment sector and role, that had cells with '?'. As such, we replace these values with Nan values. As two of these three columns, employment sector and role, had '?', around 5%, we could not simply delete them, therefore we decided changing these nan values by replacing them with the mode of individuals with the same education level.

As mentioned in the statement, only the income of people who were over 17 years old was analyzed. In this way, we eliminated from the dataset the lines where the age was less than 17 years old.

After that, we started by exploring the correlation between the metric variables and by looking at the correlation matrix we can suspect that the categorical features on our dataset are vital to predict the target income. Then decided to create a new 'Male' column that tell us whether the citizen is male or not, the information used to create this new variable was deducted by the title of the citizen's name.

Later, we analyzed how the proportion of the target variable (Income) is dispersed within each feature of each variable, by doing so, we were aiming to see their relevance when building our ML model and calculated the probability for each characteristic of each variable to be 1 and its correlations. For metric variables, we made histogram for each of those features that allowed us to get a better understanding of our data.

After analyzing the graphs made, we can notice some high values for Ticket Price (above 4000) and for Money Received (above 60000).

Since those are metric features, scaling would be an important tool to make our models perform better, since, those values, although extreme, are relevant to assess our task.

We tried to apply the IQR method to remove outliers. However, there were too many and we had little data to work with. So, as we have seen, from the different graphs we have made, that there's not a feasible cutout point to really be sure when we're dealing with an extreme value, we haven't removed values.

B. Feature design and selection

In the next step, we performed the categorical variables encoding, we started by using the one hot encoder. As we did not obtain the best results, we chosed to transform the categorical features to binary where the value 1 is obtained if the category is part of the group that has a large percentage of people with a value for income equal to 1(based on the graphs we obtained). This was particularly important in Lives with and Role.

After doing all these steps, we were left with binary features that were tested for their relevance leaving us with some core features that we feel could be relevant to the model.

C. Modelling

The F1 score is a measure of a test's accuracy and it is defined by being the harmonic mean (reciprocal of the arithmetic mean) of the precision and recall. The highest possible value of an F-score is 1, it would indicate perfect precision and recall, lowest value is 0 if either the precision or the recall is zero.

We selected the following algorithms for classification of income: Naive Bayes, Logistic Regression, k-Nearest Neighbours (kNN), Neuronal Networks, Decision Trees, Bagging Classifier, Random Forest, Adaboost Classifier, Gradient Boost Classifier and Stacking Classifier.

All the algorithms were implemented using Python's scikit-learn library.

a) Logistic Regression [1]

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm based on the concept of probability.

We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the 'Sigmoid function' or also known as the 'logistic function'.

The hypothesis of logistic regression tends it to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

We used this algorithm and different scaling approaches and did some experiments removing some features from the model.

b) K-Nearest Neighbours(kNN) [2]

Nearest-Neighbour(kNN) is a non-parametric instance-based learning method. In this case, training is not required. The algorithm begins by storing all the input feature vectors and outputs from our training set. For each unlabeled input feature vector, we find the k nearest neighbors from our training set. The notion of nearest uses Euclidean distance in the m-dimensional feature space.

Once the k nearest neighbors are selected, the predicted value will be the average of the k neighbouring outputs Before passing our data through a kNN regressor, we first Scaled on our input data. With all features normalized, each feature has a fair weight in estimating the Euclidean distance and there will be no dominating features.

c) Neuronal Networks [3]

Neural networks are computing systems vaguely inspired by the biological neural networks that constitute animal brains.

A neural network is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called *edges*.

Neurons and edges typically have a *weight* that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers.

Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

In the implementation process of this algorithm, we tunned the parameters on the model to find the best approach to the problem without overfitting it.

d) Decision Trees [4]

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

In the implementation process of this algorithm, we tunned the parameters on the model to find the best approach to the problem without overfitting it.

This ML algorithm does not need scaling to perform.

e) Bagging Classifier [5]

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (we used KNN and a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.

This algorithm encompasses several works from the literature. When random subsets of the dataset are drawn as random subsets of the samples, then this algorithm is known as Pasting. If samples are drawn with replacement, then the method is known as Bagging. When random subsets of the dataset are drawn as random subsets of the features, then the method is known as Random Subspaces. Finally, when base estimators are built on subsets of both samples and features, then the method is known as Random Patches.

f) Random Forest [6]

A random forest is a meta estimator that fits a several decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

This ML algorithm uses decision trees, therefore does not need scaling to perform. We're going to assess in this section the performance of it with different parameters to try to find the one that suits better our problem. We calculated the best number of estimators for the problem at hand.

g) Adaboost Classifier [7]

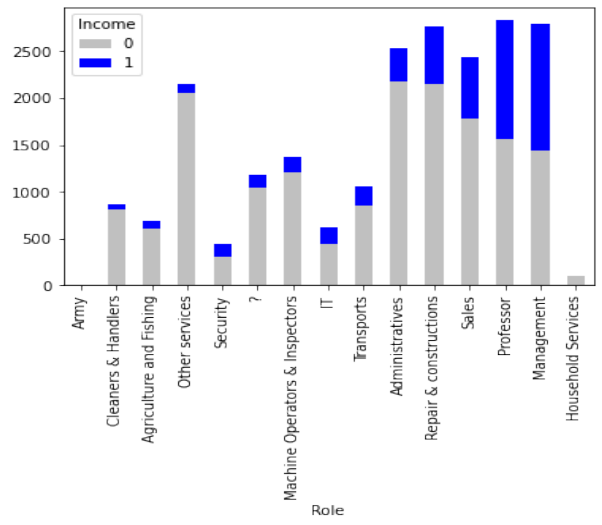
An AdaBoost classifier is a meta-estimator that fits a classifier on the original dataset and then fits repeated copies of the classifier on the same dataset but in which the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on this miss-classified cases.

h) Gradient Boost Classifier [8]

Gradient Boost, GB, builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

i) Stacking Classifier [9]

Stack of estimators with a final classifier. Stacked generalization consists in stacking the output of individual estimator and use a classifier to compute the final prediction. Stacking allows to use the strength of each individual estimator by using their output as input of a final estimator.



IV. Results

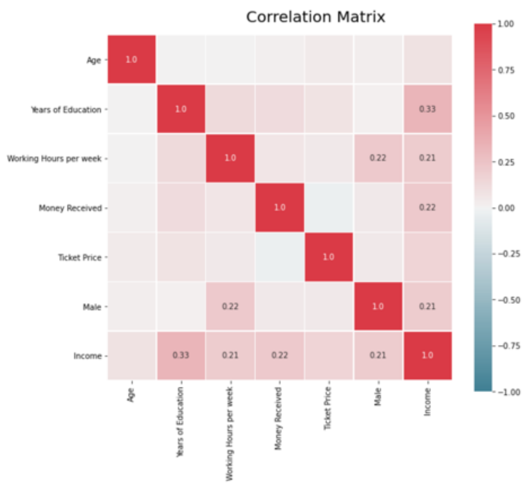
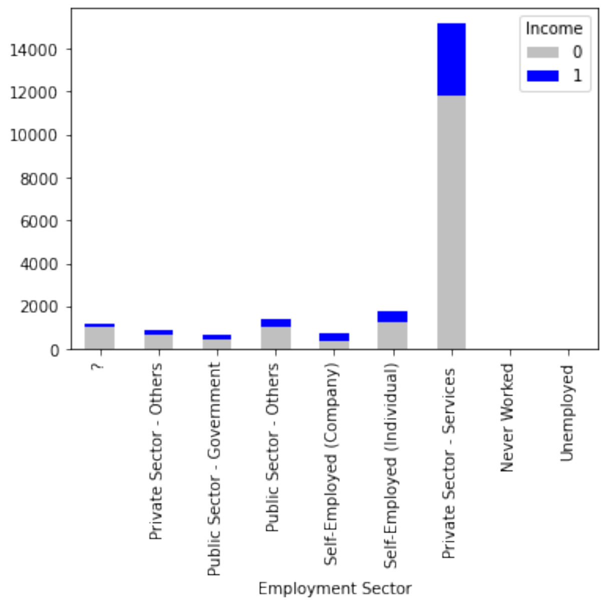
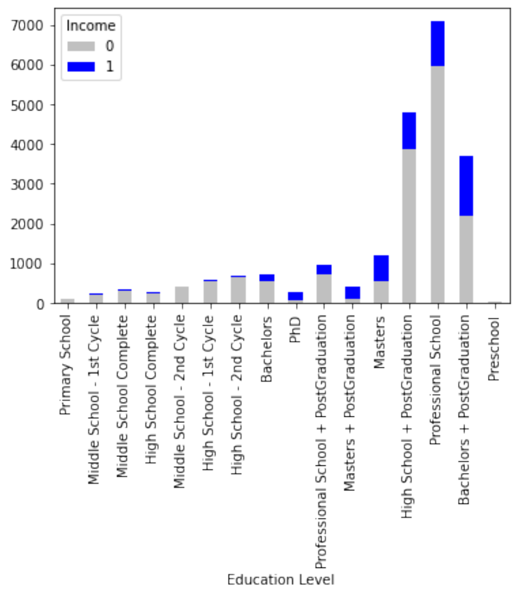
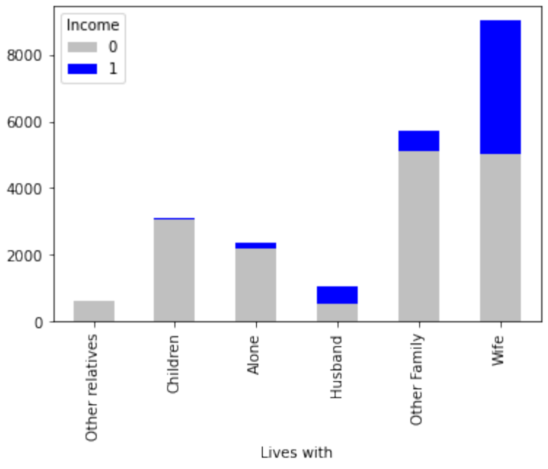
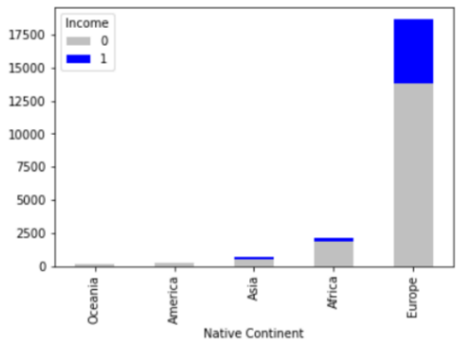


Fig 1 – correlation in between metric variables









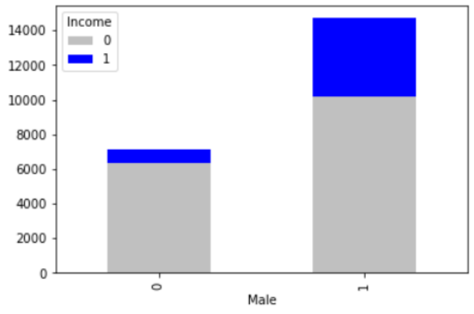


Fig 2- proportion of 1's in categorial variables

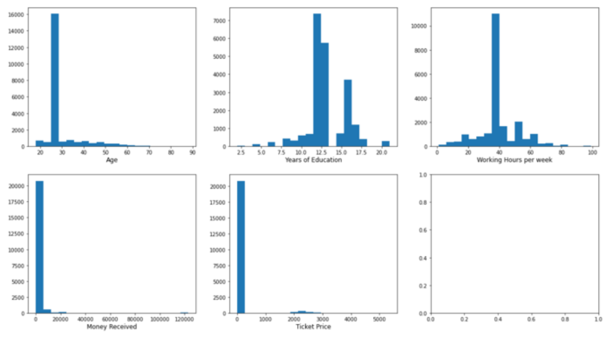
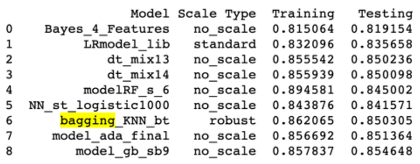


Fig 3- metric features histograms

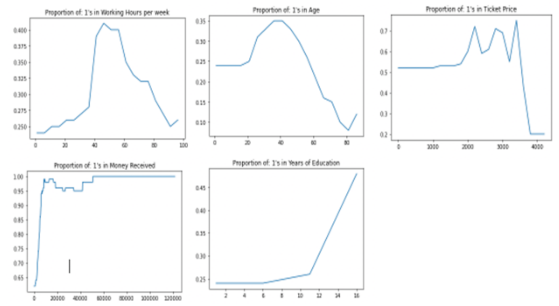
Fig 4 - proportion of 1's in metric variable

This graphics were all obtained with the purpose of trying to better understand the dataset, this way we can know how the number of 1's is distributed throughout each variable.

The results below are reported in the order based the algorithms explained above. We have all the



```
              Model Scale Type  Training   Testing
0  Bayes_4_Features   no_scale  0.815064  0.819154
1       LRmodel_lib   standard  0.832096  0.835658
2          dt_mix13   no_scale  0.855542  0.850236
3          dt_mix14   no_scale  0.855939  0.850098
4       modelRF_s_6   no_scale  0.894581  0.845002
5 NN_st_logistic1000  no_scale  0.843876  0.841571
6      bagging_KNN_bt    robust  0.862065  0.850305
7   model_ada_final   no_scale  0.856692  0.851364
8      model_gb_sb9   no_scale  0.857837  0.854648
```



models and the type of scaling that we have used and the respective score.

Tab 1 - F1-scores for the different models

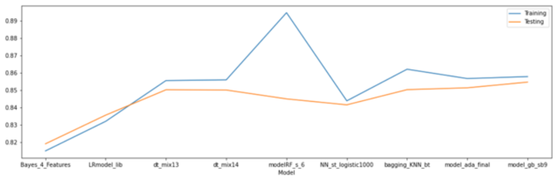


Fig 5 – Graphic with F1-scores for the different models

V. Discusion

Looking at the correlation map it is noticeable that some variables are highly correlated with the income and are not correlated with its each other therefore we can probably expect to not only have good but also meaningful variables.

After analyzing each graphic individually, we can conclude that the most relevant variables are the ones that have more noticeable proportion of 1's in one feature or that have a noticeable trend. This being Money received, hours per week, Age, Years of education and lives with. This is because they will be more impactful on the models since they hold more information.

Looking at the years of educations tables we can see that the spike on the higher frequency points matches with turning point on the proportion of 1's in income, around 11 years. To have 11 or more years of education is an important feature to have in order to achieve a 1 in income.

We can also see that the must successfully people work around 40 hours per week this tends to be also the more common number of hours to work

We tried a lot of models to see what would be "the best" one, but how can we tell which one is it? To be "the best" a model needs to have a high F-Score and a low standard deviation, in both training and test splits, in conjunction with having a low discrepancy between them to avoid overfitting.

Its noticeable that variables Ticket Price and Money Received have a lot of outliers, we decided to keep them since these points give precious information about extreme cases in either variable

Its noticeable that which the best scale type for each model their F1 score vary from 0,815 and 0,894, this is an improvement of 9%. Even though the highest F1 was 0,894 on training it had 0,845 on testing, therefore we consider it to be overfitting and don't feel like this model as potential to be considered the best one.

VI. Conclusion

During the introduction we settle to try and answer to an important question, how are we going to predict the income and in the end reach a model that predicts with a F1 score of 0,85478 in Kaggle, to reach the value we used a bagging approach consider this to be a value that allows us to really trust on our predictions.

We opt to not take into consideration the amount of computer power that it takes to run the model as one of the parameters to achieve "the best" model but in the future, it would be interesting to see if a less complex model that requires a lot less computer power could, even though lower, achieve a similar result.

Would also like to point out that there is more than likely a lot to improve and after another semester in the data science master and after having lectures about deep learning and Optimization we would probably be able to come up with an even better model.

VII. References

References

[1] Brownlee, J., 2020. *Logistic Regression For Machine Learning*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/logistic-regression-for-machine-learning/> [Accessed 27 December 2020].

[2] Brownlee, J., 2020. *Logistic Regression For Machine Learning*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/logistic-regression-for-machine-learning/> [Accessed 27 December 2020].

[3] En.wikipedia.org. 2020. *Artificial Neural Network.* [online] Available at: https://en.wikipedia.org/wiki/Artificial_neural_network> [Accessed 27 December 2020].<https://en.wikipedia.org/wiki/Artificial_neural_network> [Accessed 27 December 2020].

[4] Scikit-learn.org. 2020. *1.10. Decision Trees — Scikit-Learn 0.24.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/tree.html> [Accessed 27 December 2020].

[5] Scikit-learn.org. 2020. *Sklearn.Ensemble.Baggingclassifier — Scikit-Learn 0.24.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html> [Accessed 27 December 2020].

[6] Scikit-learn.org. 2020. *Sklearn.Ensemble.Randomforestclassifier — Scikit-Learn 0.24.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 27 December 2020].

[7] Scikit-learn.org. 2020. *Sklearn.Ensemble.Adaboostclassifier — Scikit-Learn 0.24.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html#:~:text=An%20AdaBoost%20%5B1%5D%20classifier%20is,focus%20more%20on%20difficult%20cases.x> [Accessed 27 December 2020].

[8] En.wikipedia.org. 2020. *Gradient Boosting*. [online] Available at: <https://en.wikipedia.org/wiki/Gradient_boosting> [Accessed 27 December 2020].

[9] Scikit-learn.org. 2020. *Sklearn.Ensemble.Stackingclassifier — Scikit-Learn 0.24.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingClassifier.html> [Accessed 27 December 2020].