Prediction of Adult Income: Data Mining Project Report

Jan C. Bierowiec

Contents

Chapter 1

Introduction

1.1 Overview

1.1.1 Project Description

This Data Mining project uses the real-world dataset extracted from 1994 census bureau database to predict annual incomes for adults in the United States, given a set of attributes like employment details, demographic information etc.

This code for this project was written in Python and R.

1.1.2 Summary of Results

In order to effectively use the dataset, pre-processing is required, which involves accounting for missing variables and taking into consideration an imbalanced dataset. The algorithm designed for this project is an ensemble classifier consisting of three supervised learning methods: KNN (k-nearest neighbor), Naive Bayes, and Random Forest. The bagging approach is used to balance the dataset. To impute the data, two methods are used - Random Forest and KNN - and compared to determine which is the most effective. Z-Score normalization is then applied to the algorithm to evaluate its effectiveness on the success of the algorithm compared to without normalization. It was found that which imputation method was used was insignificant to the success rate of the algorithm. However, z-score normalization had a significant effect on increasing the reliability of the algorithm.

Chapter 2

Project Details

2.1 Problem Statement

Using 1994 census bureau data, the aim is to build a predictive model that determines income level for adults. Income level is a binary target variable which indicates whether an individual makes less than or equal to $50K or greater than $50K on annual basis.

This project requires you to explore classification algorithms on a real-world dataset, and write a report explaining your experimental results. The language of implementation can be anything — the only requirement is that your program be able to interpret the given data, and be able to classify instances and produce interesting statistics.

The algorithm should be based on the classification algorithms learned during the course. Usually a straight forward implementation of one method will not lead to satisfactory performance. Also, the algorithm can be a combination of methods and should incorporate one or more data mining techniques when the situation arises. These techniques include (and certainly not limited to): – Handling imbalanced dataset – Proper imputation methods for missing values – Different treatment of various type of features: continuous, discrete, categorical, etc.

2.2 Data Exploratory Analysis

2.2.1 Variables

The census data contained fifteen variables of three distinct types: continuous, categorical and ordinal. There was only one ordinal value, education, which was originally named education-num and is described as continuous. The original education variable was deleted, making it now fourteen variables, which was recorded in string format, opting instead to use its neighboring column, education-num, as the new education variable because it translated its neighbor's string values into their corresponding integer values. Since the new education variable consisted of discrete integers whose order was determined by level of education, it was decided to treat it as an ordinal variable(i.e. "HS-grad" translated to an integer value of 9, whereas "Bachelors" translated to an integer value of 13).

Continuous Variables

- age
- fnlwgt
- age
- capital-gain
- capital-loss
- hours-per-week

Categorical Variables

- workclass
- education (education-num)
- marital-status
- occupation
- relationship
- sex

- race
- native-country

Ordinal Variables

- education-num

2.2.2 Missing Values

After a preliminary exploration of the census data, it was found that both training and testing data sets contained missing values. 3,620 out of the 48,842 instances were missing, with an additional 6 duplicate or conflicting instances. Out of these missing variables, 2,399 were found in the training data and 1,221 were found in the test data. For training and test data alike, all of the missing values were confined to three categorical values: native-country, workclass and occupation. ∼7.4% of training data instances, that is, 2399 rows, contained missing values whereas ∼7.5% of test data instances, that is, 1221 rows, contained missing values. Figure 2.1 shows the percentage of the three common missing values.

The machine learning algorithms that were decided to be implemented for this project cannot support continuous variables, only discrete/continuous variables. Therefore, the data needed to be encoded into a numerical form so that the dataset could be run through my ensemble classifier. To do this, a built-in python package called "Category-Encoders" was used, and the "Dummy-Encoder" was used to encode the data. Dummy encoding uses a new set of binary variables that is equal to the number of categories minus one (k-1). Therefore, only thirteen variables are left from the original fourteen.

2.2.3 Unbalanced Data

The training data was also found to be unbalanced with a negative skew, where 75.9% of the instances were classified as negative (<= 50,000) and only 24.1% were classified as positive (> 50,000). These are shown in the pie chart in Figure 2.2. This imbalance had to be addressed to avoid the results being skewed toward the direction of <= 50, 000, causing inaccuracy in the results. To fix the imbalance, the Bagging method was used, which involves selecting random samples of the training set and replacing them, so that a given sample may contain 0, 1, or >1, copies of the example. The learning

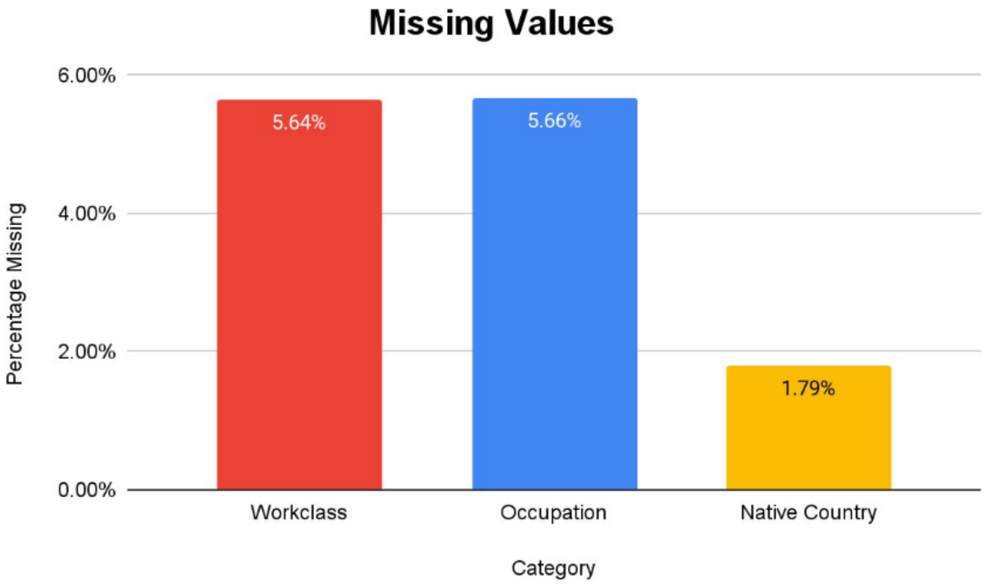

Figure 2.1: Most common missing data values

model then fits to each new sample, and the predictions made are combined to make a single prediction. Using the Bagging method helps with over-fitting and reduces the variance of decision tree based learning algorithms, and is also helpful in improving the accuracy of the Naïve Bayes and Random Forest algorithms.

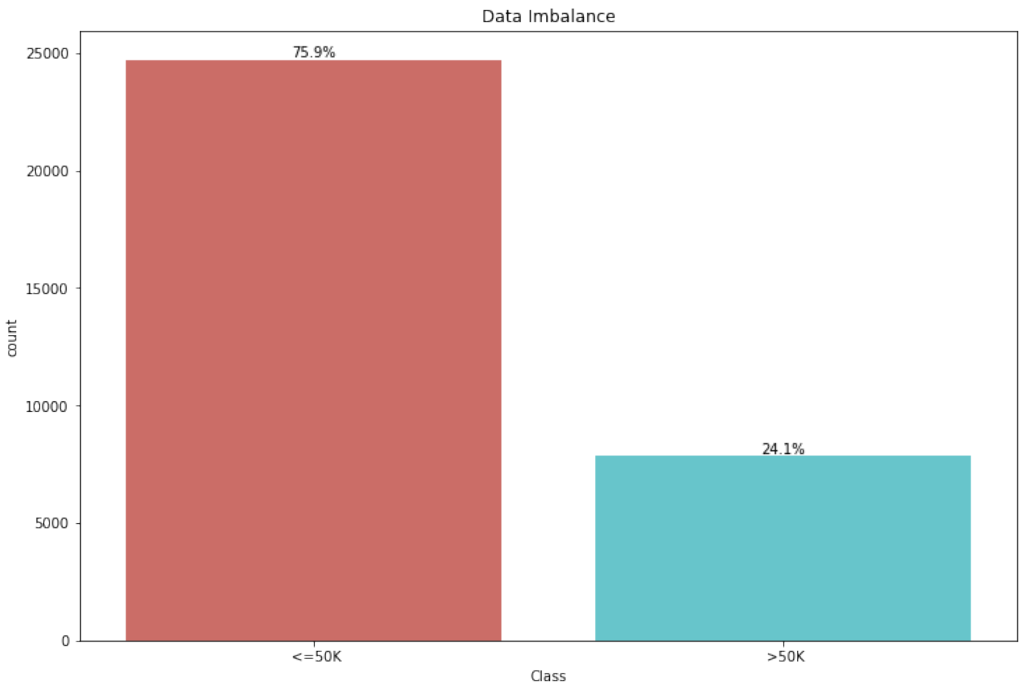

Figure 2.2: Most common missing data values

Chapter 3

Algorithm Model Development

3.1 Ensemble Models

For this project, an Ensemble Classifier was built as the predictive model. The Ensemble method approach was chosen due to the fact that it is more accurate than an individual machine learning algorithm because it is less biased towards any one algorithm. In this way, ensemble learning helps to reduce variance and over-fitting of variables, especially in relation to decision tree based algorithms. In other words, the Ensemble approach yields a robust and high performing prediction model. The Ensemble developed, three supervised learning algorithms were used, those being Naïve Bayes, KNN, and Random Forest.

The KNN, or the k-nearest neighbor algorithm uses the proximity of data points to a central point to predict the outcome of the dataset. A class label is assigned on the basis of a majority vote, i.e. the label that is most frequently represented around a given data point. The value of k can be set by the user and is usually determined by Occam's Razor, the idea that the selection which is the simplest and gives a high reliability outcome is chosen. Distance measures are determined by Euclidean distance which is given by the formula

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

.

The Naïve Bayes classifier is based on the Naïve Bayes formula which is given as

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

, and states that to find the probability of A given B, you can find the probability of B given A multiplied by the probability of A, divided by the probability of B. It is also dependent on the Naïve Bayes assumption that states each feature in a dataset is equal and independent.

The Random Forest classifier is an algorithm that utilizes many decision trees that act as an ensemble. Each tree gives a prediction and the class with the most votes becomes the overall prediction. The model is effective because the individual trees are relatively uncorrelated and so a committee will outperform the individual constituents. This helps to lessen the impact of individual error.

3.2 Imputation

To deal with the issue of missing data, it was decided to compare two models to each other. The two models chosen were the KNN based imputation and Random-Forest based imputation, both of which are model based techniques.

The KNN based approach uses an aggregation of the k values to fill in for the missing neighbors. The values were found by using the category with the most occurrences in the KNN search. In the case of a tie, one of the two would be randomly chosen. For this approach, the k value was set to be 5, which is the norm. To prepare the dataset for KNN based imputation, a built-in package of R (VIM) was used, wherein the distance is found by using the Gower's distance algorithm.

The Random Forest based approach uses multiple decision trees that contain information corresponding to attributes in the dataset. The algorithm generalizes an ensemble of these decision trees to find the most common value, which is then used for the missing value in the dataset. A built-in package of R was then used (missForest) which builds the Random Forest based on the given observations and then predicts the missing values. The algorithm repeats these steps until all of the missing values are filled.

3.3 Normaliztion

After testing the ensemble classifier to find differences in results between the KNN based imputation method and the Random Forest based imputation method, Z-score normalization was applied to the algorithm and recomputed the predictions to find its effect on accuracy measures. Z-score normalization is process of normalizing each value in a dataset such that the mean is 0 and the standard deviation is 1. It is given by the formula:

$$\text{New value} = \frac{x - \mu}{\sigma}$$

where

$$x = \text{original value}, \mu = \text{mean of data, and } \sigma = \text{standard deviation}$$

Chapter 4

Results

The results of the KNN based imputation and Random based imputation models are shown in Figure 4.1. The reliability of the predictions are given by the accuracy, prediction, recall, and F1-score measures, which are found by the confusion matrix. As we can see, the accuracy measures across both are very similar - 81.97% for KNN and 81.86% for Random Forest - such that there is no significant gain in choosing one over the other. The accuracy for the KNN imputation method is 0.11% higher than the Random Forest method, with the KNN method outperforming in all the other measures ecept for precision, where Random Forest is higher by only 0.03%.

On the other hand, it can be seen how through applying normalization to the dataset, this has a significant positive impact on the reliability of the predictions, as shown in Figure 4.2. In all measures, the predictive power is increased, with both accuracy measures for the KNN and Random Forest imputation measures increasing by ∼3.5%. As such, while the choice imputation method is insignificant to building the best machine learning model, Z-score normalization should be included in the algorithm to gain the best results.

4.1 Z-Score Normalization

In normalization, giving scores a common standard of zero mean and unity standard deviation facilitates their interpretation. This is a common procedure in statistics because values that roughly follow a standard normal distribution are easily interpretable. We use Z-score normalization which replaces the measurement unit with "number of standard deviations" away from the mean. Hence, it's a convenient tool when someone wants to compare two variables that are measured in different units.

Table 4.1 shows a direct comparison of the non-normalized and normalized KNN & Random Forest Algorithms.

Method	**Accuracy**	**Precision**	**Recall**	**F1-Score**
KNN	81.97%	32.45%	78.69%	45.95%
Random Forest	81.86%	32.48%	77.82%	45.83%
Normalized KNN	85.51%	64.12%	71.58%	67.65%
Normalized Random Forest	85.41%	61.67%	61.15%	61.41%

Table 4.1: Model Performance Summary

4.2 Future Work

In the future, it would be interesting to implement Parameter Optimization in order to find the "best" parameters for the algorithms in each ensemble classifier for each data set. Implementing a Feature Selection & Wrapper Method would also be interesting as it would help determine if removing attributes improves predictive capabilities of learning algorithm. Along with those goal, it would be good to try out a new way of balancing the data such as B(L)aging, which is short for Balanced Bagging. Last but not least, it be interesting to make use of neural networks in hopes of achieving higher performance values across the board.

4.3 Summary

In this project, a real-world dataset was evaluated and manipulated, using a number of data mining techniques, to create a predictive model for the determination of an adult's income in the US. The dataset was pre-processed to encode the data into a binary system so that it could be used for discrete/categorical models. The imbalanced dataset was attended to through the use of the bagging method to create an equal dataset. Imputation of data was compared using the KNN based approach and the Random Forest based approach, and no significant differences were noticed between them.

Z-score normalization was then applied to the algorithm and a significant improvement in the predictive power of the model was observed.

- Pantanowitz, Adam, and Tshilidzi Marwala. ”Evaluating the impact of missing data imputation through the use of the random forest algorithm.” arXiv preprint arXiv:0812.2412 (2008).

- Ganganwar, Vaishali. ”An overview of classification algorithms for imbalanced datasets.”International Journal of Emerging Technology and Advanced Engineering 2.4 (2012): 42-47.

- Stekhoven, Daniel J. ”Using the missForest package.” R package (2011): 1-11.

- Kowarik, Alexander, and Matthias Templ. ”Imputation with the R Package VIM.” Journal of statistical software 74 (2016): 1-16.

- https://towardsdatascience.com/having-an-imbalanced-dataset-here-is-how-you-can-solve-it-1640568947eb

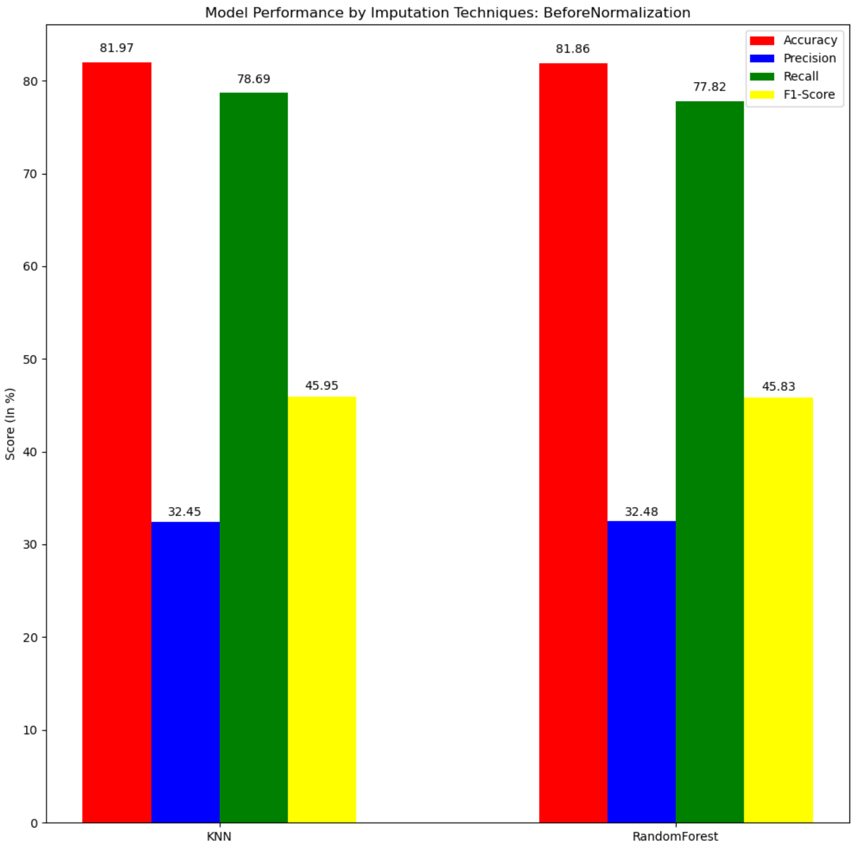

Figure 4.1: Most common missing data values

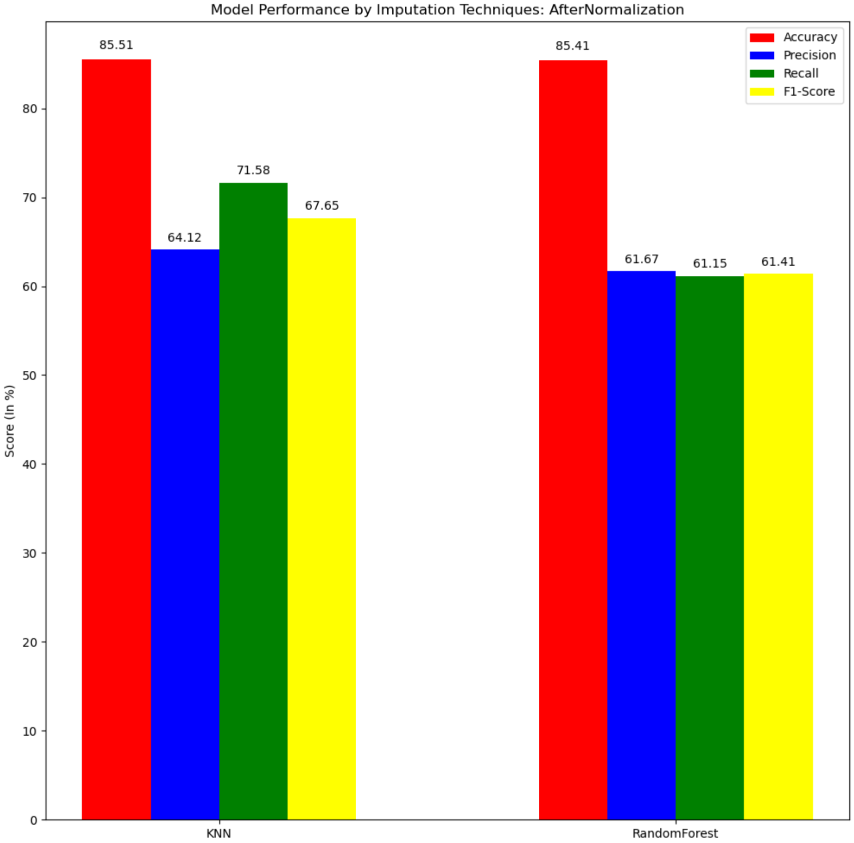

Figure 4.2: Most common missing data values