Student Number: 218713

1. Approach

1.1. Choice of Machine Learning Approach

The choice of supervised machine learning algorithm that I decided to use is called Random Forest (RF) which is a collection of multiple decision trees where each tree slightly differs from one another. This type of machine learning can be used for classification and regression tasks and can also handle numerical and categorical data which makes it ideal for this project.

1.2. How It Works

Random Forest operates by selecting random subsets of the dataset and turning them into decision trees.

A well-produced method called Bagging is used to help create these trees. Bagging involves randomly selecting subsets from the dataset and these could be chosen more than once to be included in the trees or could be never chosen. This can be good for training the algorithm as it can find patterns within the subsets it has tried but also work on new data which has not yet been tested on. [1]

Each tree calculates an output for what they predict is the correct option and then the RF selects the most popular prediction, and this is used to help train the model.

Combining all the trees together to find one output leads to improved performance, reduces the risk of overfitting and increased accuracy which is ideal for a good machine learning algorithm. [2]

1.3. Application of RF in my Project

In terms of creating a binary classification task to predict whether the person earns more or less than 50k, the selected subsets would represent each person within the dataset. The root node for each tree would be random selected, whether it be the age, education etc. Then the children node would also be randomly selected and then this continues until the tree is built.

This is repeated until there is suffice number of trees and the RF can select the most popular outcome.

2. Methods

2.1. Preprocessing the Data

To begin the process of training and testing my classifier, I have had to ensure my data was ready to be entered into the classifier; this meant having to preprocess my data.

This involved two main things, ensuring there were no white spaces in the variable names so that they are easily assessable if needed, and one hot encoding the categorical columns.

Whilst one hot encoding was not essential as RF can deal well with categorical data, there was the chance that it would help improve the performance of the classifier and reduce any errors that may have occurred. For example, there was an issue where the string could not be converted to a float when attempting to train the data so one hot encoding was one solution to that problem.

2.2. Testing and Training

To train and test my classifier, I initially split my training dataset into training and testing sets in order for the classifier to be used on unseen data which would be the testing set – I chose for 20% of the total data to be reserved for testing.

Regarding the hyper-parameters in the RF classifier used for the data, there was a lot of trial and error to see which parameters were best for the classifier to be as accurate as possible. The choice of a total of 100 trees being made and a maximum depth of 15 nodes for each tree ensured that the results were as accurate as possible whilst not taking too long for results to be generated.

The use of a seed was used for the random state parameters, and this ensures that the same result is produced each time. An advantage of this is that we can guarantee that the classifier results are reproducible, for any environment.

2.3. How performance of classifier was assessed

One way of how the classifier was assessed involves calculating a testing and training accuracy, to see how much it differs, depending on if it is on seen or unseen data.

Another way involved generating a Classification Report which can inform us on the precision, recall, F1 score and support. These are good indicators on how well the classification model can predict the outcomes. [3]

Furthermore, a Confusion Matrix was employed which can tell us how many true/false positives and true/false negatives were calculated from the classifier.

Lastly, the assessment included cross-validation and a mean cross-validation. This requires the program to do testing against different folds/subsets of the dataset to test how well the classifier can work on unseen data.

3. Results

3.1. Training and Testing Accuracy

In evaluating the performance of the classifier, we found that the accuracy of the accuracy of the training data was at a rate of 88.6% (1d.p) and accuracy of the testing data was at a rate of 86.3% (1d.p.). As both accuracies are quite similar, this meant for a good reliable classifier, especially with the testing data as this was new data that had not been seen before.

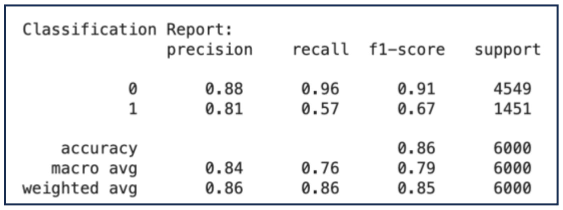
3.2. Classification Report

As seen in Figure 1, the Classification Report was relatively positive throughout.

The precision indicates there was an 88% and 81% chance of the predictions that they belonged to the correct class of earning less or or than 50k was correct, respectively.

However, for recall, the percentage of predicting the actual instances of earning more than 50k was quite low at 57%.

In summary, the weighted averages present robust results of 86% for precision, 86% for recall and 85% for the F1-Score.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.96      0.91      4549
           1       0.81      0.57      0.67      1451

    accuracy                           0.86      6000
   macro avg       0.84      0.76      0.79      6000
weighted avg       0.86      0.86      0.85      6000
```

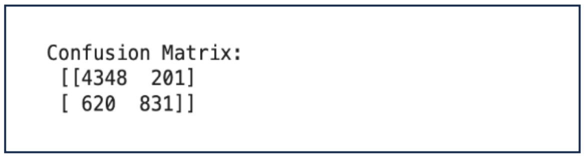
Figure 1. Classification Report of the Classifier

3.3. Confusion Matrix

In Figure 2, the Confusion Matrix created is shown.

The true positives was a total of 4348; true negatives 831; false positives 201 and false negatives 620.

Even though there are more true values than false, there are still 821 false values which means there is more room for improvement for the classifier to get an even smaller number of false values.



```
Confusion Matrix:
 [[4348  201]
 [ 620  831]]
```

Figure 2. Confusion Matrix

3.4. Cross-validation

Finally, the cross-validation results are also quite positive with them ranging from 85.3% to 86.8% and the small range implies that the classifier is good on different subsets of the dataset, across the ten folds.

The mean cross validation score is 86%, which proclaims the reliability of the classifier.

4. Conclusion

To conclude, the performance overall of the classifier is at an acceptable level but there are methods that could be done to improve performance; the confusion matrix calculating that there are 821 false negatives shows this.

There are many ways that could improve this classifier such as finding more optimal hyper-parameters using techniques such as random search or grid search. [4]

Another way involves finding the most important features of the dataset and using them to create the random forest instead, this could be done using "feature_importances_". [5]

Overall, the performance of the classifier is commendable but there are opportunities to further optimize the classifier.

References

[1] https://www.facebook.com/jason.brownlee.39 (2019). Bagging and Random Forest Ensemble Algorithms for Machine Learning. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/.

[2] Rodriguez-Galiano, V., Sanchez-Castillo, M., Chica-Olmo, M. and Chica-Rivas, M. (2015). Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. *Ore Geology Reviews*, [online] 71, pp.804–818. doi:https://doi.org/10.1016/j.oregeorev.2015.01.001.

[3] Zach (2022). *How to Interpret the Classification Report in sklearn (With Example)*. [online] Statology. Available at: https://www.statology.org/sklearn-classification-report/.

[4] Pandian, S. (2022). *A Comprehensive Guide on Hyperparameter Tuning and its Techniques*. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/.

[5] scikit-learn. (n.d.). *Feature importances with a forest of trees*. [online] Available at: https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html.