The true decision rule makes a linear comparison of incomes and classifies the middle region (between 40^{th} and 60^{th} percentiles) as true ($target = 1$). We cannot make this linear comparison in our data directly because income is not included in our feature space. But we do suspect to have features that linearly map to income, which would imply that we also need to make decision rules based on linear comparisons in our data. Particularly, we believe total expenditure -i.e. the sum of all expenditure items, which we added to the feature space- to map linearly to income. Hence, it seems prudent to opt for a class of classification models that makes decision rules based on linear comparisons of inputs: Decision Tree (DT)-based models. From these, we opt for Random Forest (RF) because of its usage of uncorrelated trees, as this prevents overfitting. A majority class prediction achieves an accuracy of 79.41% on this data. RFs are sensitive to imbalanced data, so we use the Synthetic Minority Oversampling Technique (SMOTE) algorithm by Chawla et al. (2002) to address this. Having added the total expenditure feature, we choose to remove the underlying expenditure features. This is because we assume total expenditure to capture most (if not all) of the correlation w.r.t. the target variable of the underlying expenditures. By removing them we remove the dimensionality of our feature space by about 25%.

After feature selection, there are 140 features left in the data. In a pursuit to find the RFs optimal hyperparameters for the highest accuracy out-of-sample predicting model we split the training data in a training set and a test set[1]. The training set will be used to tune the hyperparameters with a 3-fold cross-validation randomized search with 100 iterations. Bergstra and Bengio (2012) show that a random search is at least as good as grid search and more efficient. The parameter descriptions and chosen grid for our random search can be seen in table 1. Three parameters that we choose to evaluate at length are $n\_estimators$, $max\_samples$ and $max\_features$. $n\_estimators$ is evaluated between 50 and 1000. This parameter is important as it is arguably the biggest factor in the bias-variance trade-off of RFs. Hastie et al. (2009) argue that an excessively large number of trees leads to models with unnecessary variance. Therefore, we evaluate this parameter between 50 and 1000 trees. max_depth is not restricted because Segal (2004) show that restriction of tree-depth only results in small gains. Hastie et al. (2009) agrees that unrestricted tree depth does not tend to lead to excessive computational costs. Lower $max\_samples$ reduces the correlation between trees but increases the bias, as such we analyse a wide range of percentages of the full sample size as the number of $max\_samples$. We opt for the Gini-criterion as loss function as it is differentiable and less computationally expensive to compute than the cross entropy criterion. By limiting the $max\_features$ we reduce the possible additional variance in our classifier due to correlation in feature choice. However, this increases the bias in the trees. Therefore, we assess the edge scenarios with 1 feature, no restriction and the values around the most commonly used value for max features; the square root of the number of features. We have set $min\_samples\_leaf = 1$, as this is appropriate for classification. We also evaluated the k hyperparameter of SMOTE for the k-nearest neighbours. Since it is not obvious which values to evaluate for k we choose to evaluate it in a wide range: $k \in [1, 20]$, including the most commonly used value for k in SMOTE: 5.
After applying the random search we find hyperparameters in table 2 with an average validation accuracy of 82.96%. When the trained model with given hyperparameters is evaluated on the validation-set an accuracy of 82.89% is achieved. This means that it is safe to assume the random search algorithm is not overfitting as we achieve similar accuracy results out-of-sample in the test set.

[1]We split the full sample into a training set of 75% of the observations and a test set of 25%

Appendix

Table 1: The grid used for the random forest and SMOTE hyperparameter optimization.

parameter	descriptions	grid
n_estimators	Number of Trees	{50, 100,150, 200, 250, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800, 850, 950, 1000}
min_samples_leaf	Minimum number of samples required to be at a leaf node	{1}
max_samples	Maximum drawn percentage of training dataset to train each DT	{0.10, 0.20, 0.30, 0.40, 0.50, 0.60,0.70, 0.80, 0.90, 0.99999}
max_depth	Maximum depth of decision trees	{None}
max_features	Maximum number of features to consider for determination of best split	{1, 6, 11, 16, None}
criterion	Criterion used as loss function	{Gini}
k	Number of nearest neighbours in SMOTE	{1, 3, 5, 7, 20}

Table 2: Below are the optimal hyperparameters for the SMOTE algorithm and the Random Forest Model

parameter	value
n_estimators	300
min_samples_leaf	1
max_samples	0.99999
max_depth	None
max_features	1
criterion	Gini
k in knn SMOTE	3

References

[1] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization." In: *Journal of machine learning research* 13.2 (2012).

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[3] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[4] Mark R Segal. "Machine learning benchmarks and random forest regression". In: (2004).