**Abstract**

In this project, we investigated the performance of two classification models, K-nearest neighbours and decision trees, on two separate datasets, the UCI Adult dataset and the UCI balance-scale dataset, to predict if the income of an adult exceeds $50k/year based on various factors, which can be found in the UCI Adult dataset. On the adult dataset, we had to predict if the income of an adult exceeds $50k/year based on 14 factors (age, education, occupation, etc...). We achieved similar accuracy when testing on the test data on both KNN and decision tree models, with an accuracy of 86.3 % for the decision tree and 85.9% for the KNN (K=14). However, the decision tree model ran much faster than the KNN model.  
The balance-scale data was much smaller, containing only 630 rows, which affected the performance of our model on unseen data as it was overfitting on the training data. We achieved high accuracies in cross-validation (84.6% for KNN, and 76.9% on the decision tree), however, on unseen data, the KNN model was only 37% accurate, and the decision tree was 47% accurate. This was a bit surprising, as usually Decision trees tend to overfit the data more than KNN, but in this case, the opposite is true.

**Introduction**

The first task was to load the datasets into a pandas data frame, drop the unwanted features, and also correct the missing values. We had two options for dealing with NAs, dropping all rows with missing values, or replacing missing values by the mean (for continuous variables) or the mode (for discrete variables) of the feature. We found that replacing the NAs led to improved results, since we are not losing valuable information.  
The next step was to split the dataset into continuous and discrete variables, and the label column, then perform one-hot encoding on the discrete variables, and group the continuous and encoded discrete variables into one NumPy array, this will be our X, the label will be Y.  
Finally, we perform 5-fold-cross-validation on both models with X and Y as input, to find out what the best hyperparameters are, and to see the accuracy results.  
To perform cross-validation, we wrote a function for each model, that work in the following way:  
input: X and Y (where Y is the label for each X), and a list of hyperparameter values to test (e.g. K=[1,2,3] will run cross validation with K=1,2,3)  
returns: list of hyperparameter values with associated accuracy  
Implementation: First, we split X and Y into L parts, and take one part for validation, and L-1 for training. Then we return the performance with the chosen hyperparameters. The performance is the sum of accurate predictions from the validation set divided by the size of the validation set.

**Datasets**

For both datasets, we processed them into pandas data frames, and did manipulations on the data frame. Instead of dropping all rows with NA’s, and losing valuable data, we decided to replace them with the mean or mode (as described in the introduction), as it gives a good estimate for the feature, and does not waste the other features in the row that are not missing. This led to improved results.  
  
**Dataset 1: Adult dataset**  
This dataset contains 15 columns, 14 features, and 1 label column, and 32,560 rows. We dropped the fnlwgt feature, as it is noisy to KNN and irrelevant. We also dropped the education feature as it is a duplicate to education-num (there is a one-to-one mapping with the two), and we decided to keep education-num and not education since the former is an integer, and gives higher scale to instances with higher education, which should be an important factor in predicting income.  
We addressed missing data by first replacing the “?” values by numpy.nan values, and then calling the mode or mean function (depending if the feature is discrete is continuous), that replaces the nan values by the mode and mean respectively. This improved the results, because there are over 1000 rows with missing values, so removing all of them would result in a significant loss of information.  
The full data frame along with the relevant information can be seen in the jupyter notebook.

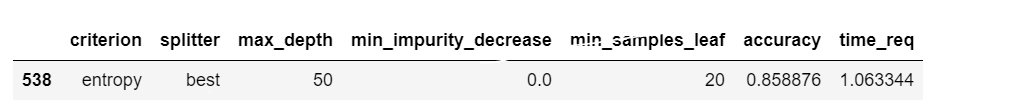
**Dataset 2:**

**Results**

1. We noticed that both models give similar accuracies, but decision trees run much faster, and give slightly better results, which makes them more suited to this dataset.   
   Running 5-fold cross-validation on KNN with 20 values for k took 335 seconds, and the best accuracy was 85.4% for k=14.  
   Running 5-fold cross-validation on the decision tree with 720 permutations of hyperparameters took 330 seconds only, and the best accuracy was 85.9%. This means the decision tree was over 36 times faster than the KNN model and yielded better accuracy.
2. - For KNN, we could tune the values of K, or p (the norm used to measure distance). We noticed that using the l-1 norm takes more time than the l-2 norm (strangely), doubles the computational complexity, and doesn’t improve performance, so we stuck with the l-2 norm  
   The values for K didn’t make a vast difference, and only affected performance by a few percentage points, all values were yielding results between 79%-85%.  
   - For decision trees, there were a lot of hyperparameters to tune, and a wide range of values each hyperparameter can have, so we stuck only to the ones we thought were most relevant (criterion, splitter, max\_depth, min\_impurity\_decrease, and min\_samples\_leaf).   
   A surprising result was that choosing min\_impurity\_decrease=0 leads to better results on both datasets, since we thought this would be overfitting to the training data. A max\_depth of 10 was optimal, which is reasonable, Figures 1 and 2 shows the best hyperparameters we found.  
   - Another surprising result is that scaling had negligible, or negative, effects on the accuracy of the KNN model on both datasets. We expected that scaling more important features (such as education and age) would lead the model to emphasize them more, and thus improving the results, but this was not the case. Figures 3 and 4 show the results achieved by scaling for each dataset.
3. Reducing the amount of data by removing all rows with missing values, severely reduced the accuracy of both models, which is why we decided to replace the missing values instead of removing them. It is not surprising that this had a positive effect, since we give more data to the model to learn from, allowing it to give better predictions.  
   More significantly, the models applied to the balance-scale dataset, which contained only 623 rows compared to the 32,560 on the adult dataset, overfitted to the training data and performed very poorly on unseen data. Which shows that the size of the dataset plays a very important role in achieving high results, and there is a direct correlation between the two: more training data leads to better accuracy on unseen data.

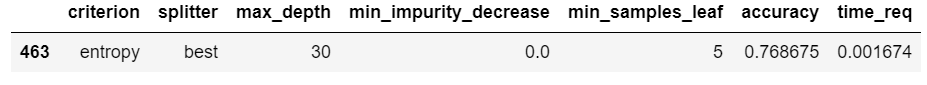
**Appendix**

**Figure 1:**

****

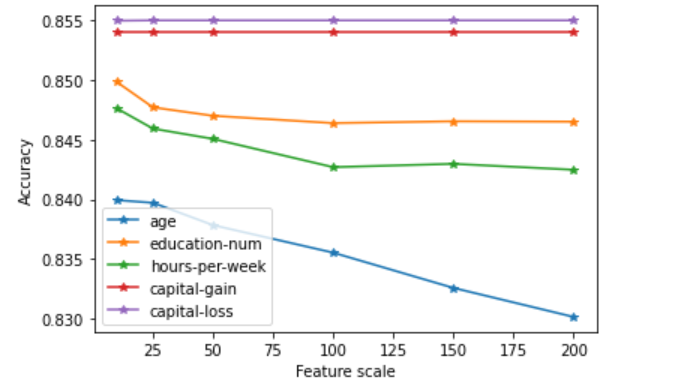
Title: Hyperparameters that achieved the highest accuracy in the cross validation for the decision tree model on the adult dataset.

**Figure 2:**

****

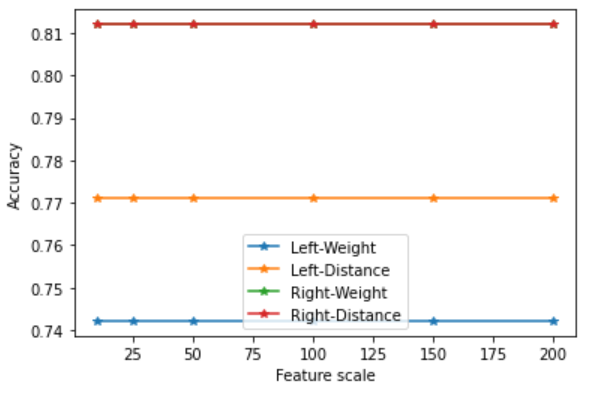
Title: Hyperparameters that achieved the highest accuracy in the cross validation for the decision tree model on the adult dataset.

**Figure 3:**

****

Title: Plot showing the accuracy for KNN (K=14) with different scaling of parameters on the adult dataset

**Figure 4:**

****

Title: Plot showing the accuracy for KNN (K=13) with different scaling of parameters on the scale-balance dataset

N.B: We only used the best K from the cross validation when scaling as it would be too time consuming to try all possible K values, even though it might lead to better results.