Gün Kaynar
22101351
Part 1

In this part, I first started with loading the Thyroid data set. I have used Pandas library along with numpy to get features and labels of our dataset into 2 dimensional arrays.
I have split my training data into cross-validation and training set as %20 being cv %80 being training. I have used Decision Tree Classifier model from Scikit-learn library of Python. I have prepared a hyperparameter space to pick the best hyperparameters using cross validation set. The hyperparameters were as follows:
criterion = ["gini", "entropy"]
splitter = ["best","random"]
max_depth = [10, 15, 25, 30]
min_samples_split = [5, 10, 15]
min_samples_leaf = [2, 10, 20]
max_features = ["auto", "sqrt", "log2"]

After fitting the cross validation set for those hyperparameter set, the best ones for me were:
{'criterion': 'gini', 'max_depth': 25, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}

Then I have trained the model using my training set. I have drawn the Decision Tree using Scikit-learn library's plot function. And it can be seen in folder q1, under the name 000decision_tree.pdf

I have then calculated the accuracy for each class and confusion matrix on my training set. The class-based accuracies were found as follows:
Class 1: 1.00
Class 2: 1.00
Class 3: 0.98

Where the confusion matrix has been as below:

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	76	0	0
Actual Class 2	0	154	0
Actual Class 3	10	48	2729

After that, I have tested my model on my test set and calculated the class-based accuracies.
Class 1: 0.89
Class 2: 0.86
Class 3: 0.97

The confusion matrix drawn on test-set can be seen below:

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	65	0	8
Actual Class 2	0	152	25
Actual Class 3	41	67	3070

As can be seen from the difference between accuracies of training and test sets, our model has overfit to training set, i.e., there is a variance problem.

To solve this overfit problem, I have applied pruning to my tree. I have chosen Minimal Cost-Complexity Pruning and used alpha = 0.005 for pruning. Then, I have again used my cross-validation set to find the best parameters and this time the best hyperparameters have been:
{'ccp_alpha': 0.005, 'criterion': 'entropy', 'max_depth': 15, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}

I have drawn the Decision Tree using Scikit-learn library's plot function. And it can be seen in folder q1, under the name 010decision_tree.pdf

On training set class-based accuracies have been:
Class 1: 1.00
Class 2: 1.00
Class 3: 0.98

And the confusion matrix was drawn as follows:

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	76	0	0
Actual Class 2	0	154	0
Actual Class 3	47	3	2737

On test set, the class-based accuracies have been calculated as:
Class 1: 1.00
Class 2: 1.00
Class 3: 0.97

And the confusion matrix for test set has been:

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	73	0	0
Actual Class 2	0	177	0
Actual Class 3	75	13	3090

We can conclude that the Minimal Cost-Complexity Pruning has helped us overcome the overfitting in our model, because we can see from the accuracies of training and test sets that there is not a big difference between them, thus we have a less variance.

Later, I have applied min-max normalization on training, cross-validation and test sets. The results have shown that the normalization did not change anything on my Decision Tree classifier model. The class-based accuracies on training set were:
Class 1: 1.00
Class 2: 1.00
Class 3: 0.99
Whereas on test set, the accuracies have been calculated as:
Class 1: 0.88

Class 2: 0.84
Class 3: 0.97

I have drawn the Decision Tree using Scikit-learn library's plot function. And it can be seen in folder q1, under the name 100decision_tree.pdf

Finally, I have applied oversampling and undersampling on training set to overcome the imbalance in class distributions. The table below shows the training and test accuracies when oversampling, undersampling and no balancing are used.

	No balancing	Oversampling	Undersampling
Training set class-based accuracies	Class 1: 1.00 Class 2: 1.00 Class 3: 0.98	Class 1: 1.00 Class 2: 1.00 Class 3: 0.99	Class 1: 1.00 Class 2: 0.93 Class 3: 0.88
Test set class-based accuracies	Class 1: 0.89 Class 2: 0.86 Class 3: 0.97	Class 1: 0.90 Class 2: 0.98 Class 3: 0.98	Class 1: 1.00 Class 2: 0.76 Class 3: 0.84

When I balanced the training samples by oversampling the minority groups, my test set accuracies have increased, on the other hand undersampling seemed not to increase the test accuracies but decrease the training accuracy. This is probably because that the variance in the training data was not detained since we delete the majority group's samples.
I have drawn the Decision Tree using Scikit-learn library's plot function. And it can be seen in folder q1, under the name 001decision_tree.pdf for oversampling and 002decision_tree.pdf for undersampling.

Part 2
I have developed a Decision Tree classifier model that uses entropy as the impurity measure and uses maximum depth and minimum sample number to split as prepruning technique. I have again split my training data into cross-validation and training sets. Then, I have used cross-validation accuracy to decide on the values of maximum depth and minimum sample to split parameters. The best parameters have been max_depth=10, min_samples_split=5.

I have calculated the training and test set class-based accuracies as follows:

Training set class-based accuracies	Test set class-based accuracies
Class 1: 0.98 Class 2: 1.00 Class 3: 0.99	Class 1: 0.96 Class 2: 0.99 Class 3: 0.99

The confusion matrix on training set is shown below.

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	91	0	2
Actual Class 2	0	191	0
Actual Class 3	0	1	3487

The confusion matrix on test set is shown below.

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	70	0	3
Actual Class 2	0	176	1
Actual Class 3	7	11	3160

My decision tree model has been drawn below:

```
Split: X16 <= 0.006
        Leaf: 3
        Split: X20 <= 0.064
                Split: X7 <= 0.0
                        Split: X17 <= 0.022
                                Leaf: 1
                                Split: X16 <= 0.014
                                        Leaf: 3
                                        Leaf: 1
                        Leaf: 3
                Split: X2 <= 0.0
                        Split: X18 <= 0.148
                                Split: X7 <= 0.0
                                        Split: X19 <= 0.047
                                                Leaf: 3
                                        Leaf: 3
                                Leaf: 3
                        Leaf: 3
```

Part 3

Lastly, I have loaded cost data for feature splitting. I have embedded the feature costs into my Decision Tree Classifier and changed my splitting method. Now, my model not only considers the information gain (entropy difference), but it also considers the cost of each feature. I have calculated the criterion to split as:

$$Criterion = Information\ gain \times beta - Cost$$

I have selected beta parameter as [100, 500, 1000] and learnt it from cross-validation set. The best beta has been 500 along with other hyperparameters: max_depth=10, min_samples_split=5.

I have calculated the training and test set class-based accuracies as follows:

Training set class-based accuracies	Test set class-based accuracies
Class 1: 0.98 Class 2: 1.00 Class 3: 1.00	Class 1: 0.96 Class 2: 0.99 Class 3: 0.99

The confusion matrix on training set is shown below.

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	91	0	2
Actual Class 2	0	191	0
Actual Class 3	0	0	3488

The confusion matrix on test set is shown below.

	Predicted Class 1	Predicted Class 2	Predicted Class 3
Actual Class 1	70	0	3
Actual Class 2	0	175	2
Actual Class 3	7	11	3160

I have computed the cost of classifying each sample in the test set. Then I have averaged each class's computing cost:

Average cost for classifying class 1	Average cost for classifying class 2	Average cost for classifying class 3
66.02	55.55	23.46

```
Split: X16 <= 0.006 with cost= 22.78
        Leaf: 3
        Split: X20 <= 0.064 with cost= 25.92
                Split: X7 <= 0.0 with cost= 1.0
                        Split: X17 <= 0.022 with cost= 11.41
                                Leaf: 1
                                Split: X16 <= 0.014 with cost= 22.78
                                        Leaf: 3
                                        Leaf: 1
                        Leaf: 3
                Split: X2 <= 0.0 with cost= 1.0
                        Split: X18 <= 0.148 with cost= 0.0
                                Split: X7 <= 0.0 with cost= 1.0
                                        Split: X19 <= 0.047 with cost= 0.0
                                                Leaf: 3
                                                Split: X0 <= 0.78 with cost= 1.0
                                                        Leaf: 2
                                                        Split: X17 <= 0.0208 with cost= 11.41
                                                                Leaf: 2
                                                                Leaf: 3
                                        Leaf: 3
                                Leaf: 3
                        Leaf: 3
```