1.1) Implement the ID3 algorithm from scratch

The learn and predict functions are created. In the learn function y is considered to be in the last column of X.

Learn uses the functions: **grow_tree**, and **prune_tree** implemented in **MyDT.py** file.

Predict uses **Tree_Predict** function implemented in **MyDT.py** file.

List of implemented functions and classes in **MyDT.py** file:

Functions	**Classes**
Gini_index	End_Node
Entropy	Node
purity	Criterion
divide	
find_best_attribute	
grow_tree	
Tree_Predict	
tree_visualization	
accuracy_for_a_set	
prune_tree	

1.2) Add Gini index

The purity (info gain) can also be calculated with the Gini index. The Gini index of a vector is calculated with **Gini_index()** function. **purity()** uses the **Gini_index()** to calculate the purity of information gain based on this measure.

1.3) Add reduced-error pruning

It is added to the **learn()** function in main. When the **prune=True** it first learns a full tree by using **grow_tree()**, then it prunes the tree by **prune_tree()**. The indexes of the training data that should be used for pruning are being passed to the **lean()**.

1.4) Evaluate your algorithm

The specified dataset is loaded in **Main.py** or **Main.ipynb**. Model selection for the algorithm is skipped due to limitations in time for delivering the project. However, through several runs manually the following results are received:

Train_set size/(Test_set size + Train_set size) = .7
Train_set after pruning size/(pruning_set size + Train_set after pruning size) = .6
One time run:

	prune=False	prune=True
Gini	Time elapsed for learning: 4.957105200000115 Accuracy : 0.19617224880382775	Time elapsed for learning: 3.530393500000173 Accuracy : 0.22009569377990432
entropy	Time elapsed for learning: 5.963340000000244 Accuracy : 0.17862838915470494	Time elapsed for learning: 4.125450800000181 Accuracy : 0.19856459330143542

With DecisionTreeClassifier from sklearn (default values):
Time elapsed for learning: 0.020507899999756773
Accuracy : 0.19218500797448165

1.5) Compare to an existing implementation

With the provided results from one run (due to limitations in time, i didn't du the test several times and average), we can conclude that the accuracy is comparable with the learned model based on the implemented algorithm. However, regarding the speed of learning, learning a model with the implemented algorithm is much slower.

Differences in the learned models and the speed of learning between the implemented algorithm and the sklearn function are resealable, since:

- parameters are different
- The discretization of the continuous variables can be different. In the implemented algorithms the intervals of input values for each feature are discretized to two sets in each level, and the considered threshold is the middle point of the range of the values in that attribute of the set.

The **Main.py** or **Main.ipynb** files are the same and they are the main ones. All the needed functions and classes implemented in **MyDT.py** are imported in the main file by: **from MyDT import *.** For more information please check the notebook; more explanation can be provided in the case.