Lab 2: Decision Tree with scikit-learn

Student Name: Châu Tấn Kiệt

Student ID: 21127329

Student Class: 21CLC04

Self-evaluation:

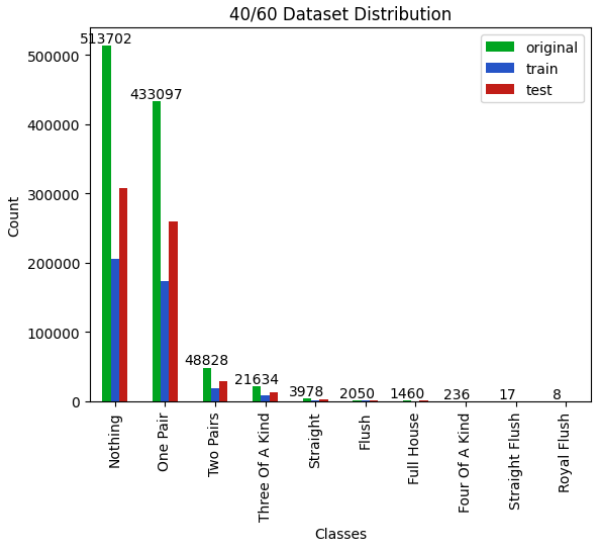
No.	Specifications	Status
1	Preparing the Data sets	Completed
2	Building the decision tree classifiers	Completed
3	Evaluating the decision tree classifiers	Completed
	Classification report and confusion matrix	Completed
	Comments	Completed
4	The depth and accuracy of a decision tree	Completed
	Trees, tables, and charts	Completed
	Comments	Completed

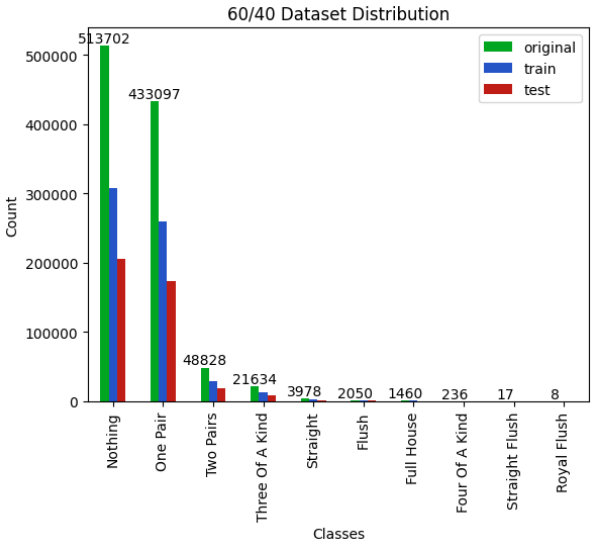
Preparing the Data sets

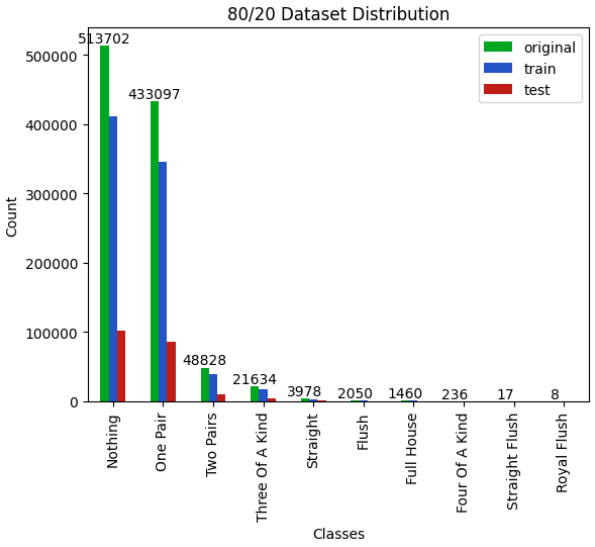
There are four subsets from the merged data in `poker-hand-data.csv`:

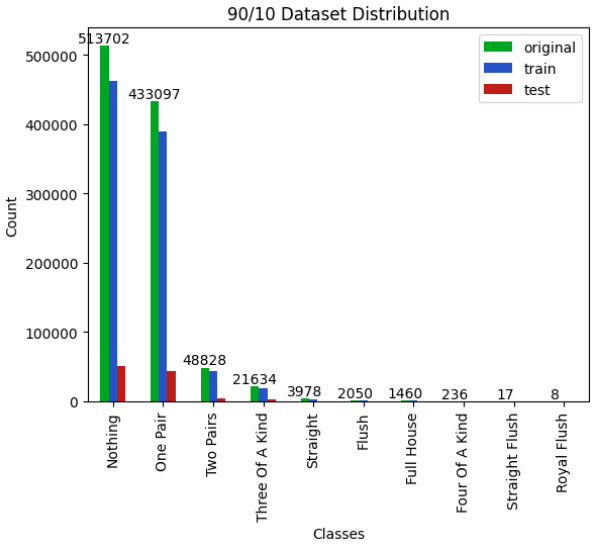
- `feature_train`: a set of training examples, each of which is a tuple of 42 attribute values (target attribute excluded).
- `label_train`: a set of labels corresponding to the examples in `feature_train`.
- `feature_test`: a set of test examples, it is of a similar structure to `feature_train`
- `label_test`: a set of labels corresponding to the examples in `feature_test`

The Jupyter Notebook provided shows the dataset visualization for sets of different proportions, including 40/60, 60/40, 80/20, and 90/10 for (train/test). Here's the results:









Building the Decision tree classifiers

For each set of different proportions, I fit an instance of `sklearn.tree.DecisionTreeClassifier` (with `log_loss` , which is equal to information gain) to each training set and visualize the resulting decision tree using `graphviz`.

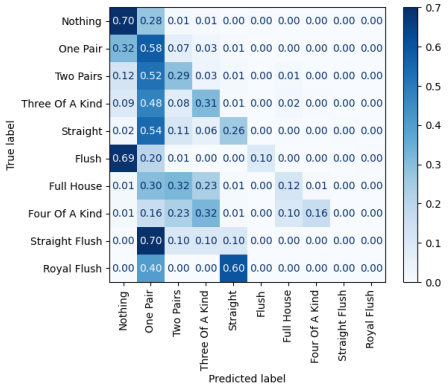
The Jupyter Notebook provided displays the visualization of the dataset for the resulting decision tree using different proportions of 40/60, 60/40, 80/20, and 90/10 for train/test sets.

Evaluating the decision tree classifiers

The Jupyter Notebook provided prediction for the examples in the corresponding test set and displays the classification report and confusion matrix for a classifier on the data set.

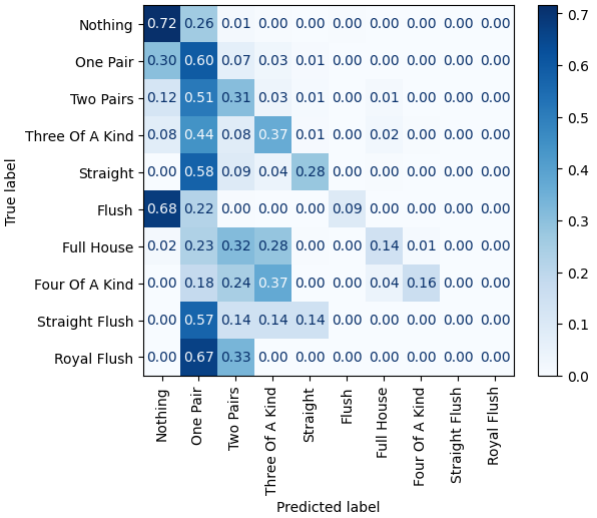
40/60:

	precision	recall	f1-score	support
Nothing	0.71	0.70	0.70	308221
One Pair	0.58	0.58	0.58	259858
Two Pairs	0.27	0.29	0.28	29297
Three Of A Kind	0.28	0.31	0.30	12980
Straight	0.22	0.26	0.24	2387
Flush	0.06	0.10	0.07	1230
Full House	0.10	0.12	0.11	876
Four Of A Kind	0.13	0.16	0.14	142
Straight Flush	0.00	0.00	0.00	10
Royal Flush	0.00	0.00	0.00	5
accuracy			0.62	615006
macro avg	0.23	0.25	0.24	615006
weighted avg	0.62	0.62	0.62	615006



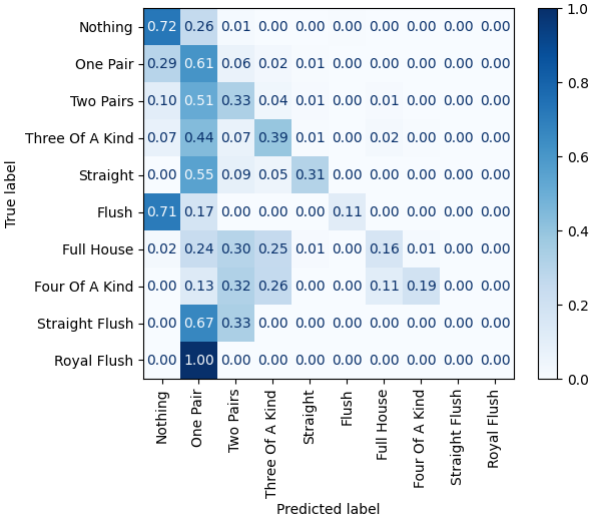
60/40:

	precision	recall	f1-score	support
Nothing	0.72	0.72	0.72	205481
One Pair	0.60	0.60	0.60	173239
Two Pairs	0.29	0.31	0.30	19531
Three Of A Kind	0.33	0.37	0.35	8654
Straight	0.25	0.28	0.26	1591
Flush	0.06	0.09	0.07	820
Full House	0.11	0.14	0.13	584
Four Of A Kind	0.15	0.16	0.16	94
Straight Flush	0.00	0.00	0.00	7
Royal Flush	0.00	0.00	0.00	3
accuracy			0.63	410004
macro avg	0.25	0.27	0.26	410004
weighted avg	0.64	0.63	0.64	410004



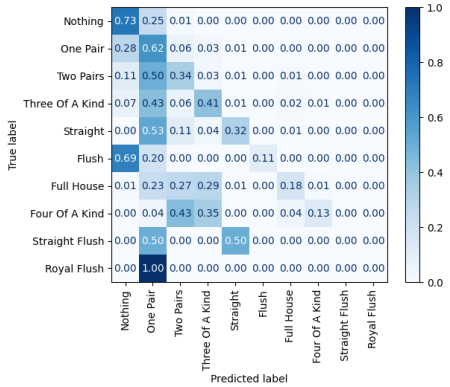
80/20:

	precision	recall	f1-score	support
Nothing	0.73	0.72	0.73	102740
One Pair	0.61	0.61	0.61	86619
Two Pairs	0.31	0.33	0.32	9766
Three Of A Kind	0.36	0.39	0.38	4327
Straight	0.27	0.31	0.29	796
Flush	0.08	0.11	0.09	410
Full House	0.15	0.16	0.15	292
Four Of A Kind	0.13	0.19	0.16	47
Straight Flush	0.00	0.00	0.00	3
Royal Flush	0.00	0.00	0.00	2
accuracy			0.65	205002
macro avg	0.26	0.28	0.27	205002
weighted avg	0.65	0.65	0.65	205002



90/10:

	precision	recall	f1-score	support
Nothing	0.74	0.73	0.74	51370
One Pair	0.62	0.62	0.62	43310
Two Pairs	0.32	0.34	0.33	4883
Three Of A Kind	0.37	0.41	0.39	2163
Straight	0.30	0.32	0.31	398
Flush	0.07	0.11	0.09	205
Full House	0.14	0.18	0.16	146
Four Of A Kind	0.09	0.13	0.11	23
Straight Flush	0.00	0.00	0.00	2
Royal Flush	0.00	0.00	0.00	1
accuracy			0.66	102501
macro avg	0.27	0.28	0.27	102501
weighted avg	0.66	0.66	0.66	102501



The classification report and confusion matrix indicate that the accuracy of the test proportion increases as the train proportion gets larger. For example, the accuracy increases from 0.62 for the 40/60 set to 0.66 for the 90/10 set.

The depth and accuracy of a decision tree

The provided Jupyter Notebook displays the decision tree generated by `graphviz` for different `max_depth` values, ranging from None to 2, 3, 4, 5, 6, and 7.

The results in this table are rounded up to 6 decimal places.

max_depth	None	2	3	4	5	6	7
Accuracy	0.646633	0.505956	0.508283	0.526785	0.557629	0.557628	0.557653

Based on the results, it is clear that the accuracy changes drastically at first. However, as the `max_depth` increases, the changes in accuracy become much slower.