ML Mini Project via Orange: Hyperparameter Tuning

Name: Waris Damkham ID: 6388014

Part 1

1) Use Orange to build Decision Tree model for Heart Disease dataset (search "Heart Disease" from Dataset icon).

2) From Tree you are to evaluate using Test and Score

3) Now try various parameter setting for the tree model for prediction task as indicate in the "Model" column of the table below. Select the "best" scoring tree.

Table 1: Finding best parameters via hyperparameter tuning experiment.

No	Model	Accuracy (CA)	F1 (class 1)	Precision (class 1)	Recall (class 1)
1	DT (leave=2, split=5, depth=100), (Train)	0.944	0.938	0.955	0.921
2	DT (leave=10, split=5, depth=100), (Train)	0.848	0.827	0.866	0.791
3	DT (leave=10, split=30, depth=100), (Train)	0.848	0.827	0.866	0.791
4	DT (leave=2, split=5, depth=3), (Train)	0.848	0.827	0.866	0.791

Part 2

1) Use Orange to build Decision Tree model for Iris dataset (search "Iris" from Dataset icon)

2) You are to use Tree model in orange and try to view the resulting tree via "Tree Viewer"

3) Explore the parameter setting that you can for the "best" looking tree in your opinion. In the space below explain why you think this is the "best" tree and what are your settings?

Setting

- **Induce binary tree**: Enable

- **Min. number of instances in leaves**: 2

- **Do not split subsets smaller than**: 5

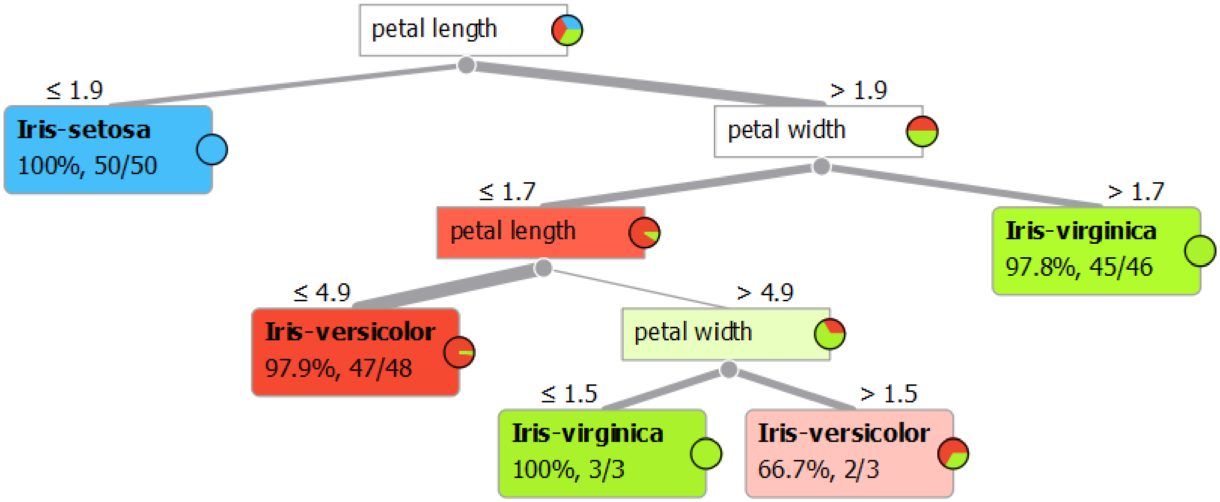
- **Limit the maximal tree depth:** 100

- **Stop when majority reaches [%]**: 95%

Why do I think this is the "best" tree?

Ans: In this setting, it will display the best-scoring tree, and after using the tree viewer, the tree is easiest to read and gives more clearly defined information.

4) Download the image of your best-looking tree from Tree Viewer and insert it in the box.



Part 3:

1) Use Orange to create Linear regression model for HDI dataset (search “HDI” from Dataset icon)

2) Use Test and Score to experiment using the following methods.

a. *Test on training data*

b. *Cross Validation (5-fold)*

3) Adjust various Regularization setting using the values indicated in the “Model” column of the table below. Record the resulting RMSE on each experimental methods in the table.

Table 2: Test on Train RMSE vs. CV RMSE

No	Model	RMSE (Train)	RMSE (CV)
1	LR	0.022	0.046
2	LR (L2, alpha=0.01)	0.022	0.045
3	LR (L2, alpha=0.16)	0.022	0.046
4	LR (L2, alpha=1.00)	0.022	0.047
5	LR (L1, alpha=1.0)	0.067	0.117
6	LR (ElasticNet, alpha=1.0)	0.052	0.122

4) Discuss in the box below if you can detect “underfitting” and/or “overfitting” using the results from Table 2.

From the results of the second table, nothing can be detected by us. We can't see "underfitting" or "overfitting" from the RMSE value because, to know "underfitting" or "overfitting," we have to look at the total value of the linear regression. The second table, on the other hand, shows that the model predicts that the y-axis value will move positive and negative, but not more than the figure above, and that the closer to 0, the more accurate the prediction, but also the more difficult it, because it may lead to overfitting problems.

Part 4:

1) Use Orange and Wine Quality Red (search "Wine Quality - Red" from Dataset icon) to find the best parameter setting for three classifiers (listed below). First, you are to split the data into training set and testing data set using the 80% of the data for training set. Then, you must use the concept of tuning dataset for the hyperparameter tuning experiment to find the best parameters. However, you can decide the search space for the hyper-parameter combinations on your own.

1. Random Forest
2. Linear Regression
3. kNN

2) Compare the tuning using the single split VS the 5-fold cross validation. For the first, further split 80% of the training data into train and tune set. Then, train the model on train and test on the tune. Record the data in RMSE (Train). For RMSE (CV), simply perform 5-fold CV on the training data.

Table 3: Tuning Experiments

No	Model	Best-parameters	RMSE (Train)	RMSE (CV)
1.1	Random Forest	0.622	0.281	0.622
1.2	Random Forest (5 CV)	0.638	0.294	0.638
2.1	Linear Regression	0.793	0.792	0.793
2.2	Linear Regression(5 CV)	0.795	0.793	0.795
3.1	kNN	0.756	0.603	0.756
3.2	kNN (5 CV)	0.777	0.612	0.777

3) Pick one best hyper-parameters of each model for RMSE (Train) and RMSE (CV), and train the model to test on the test data (initial 20% split). Record the results below

Table 4: Final results

No	**Model**	**Best-parameters From RMSE (Train)**	**RMSE (Test)**
1	Random Forest	0.294	0.603
2	Linear Regression	0.793	0.655
3	kNN	0.612	0.750

No	**Model**	**Best-parameters From RMSE (CV)**	**RMSE (Test)**
1	Random Forest	0.638	0.603
2	Linear Regression	0.795	0.655
3	kNN	0.777	0.750

4) Compare and contrast the results from two tuning experiments below.

From the above table results, the data split of 80% for training and 20% for testing is better and more accurate than 5-fold cross-validation in the test of testing data. But in the training and cross-validation tests, it was found that data splitting using cross-validation produced better and more accurate results.