Data Mining Project 2

P76061425 林聖軒

Environment

- DISTRIB_ID=Ubuntu
- DISTRIB_RELEASE=18.04
- DISTRIB_CODENAME=bionic
- DISTRIB_DESCRIPTION=Ubuntu 18.04.1 LTS

Usage

Classifier

```
$ python3 classifier.py [-h]
```

optional Options	**Description**
-h --help	show this help message and exit
-m METHOD	Classification method, dct=(Decision Tree),svm=(Support Vector Machine), default = dct
-train, TRAIN_PATH	Input training data file, default = ./data/train_data.txt
-test TEST_PATH	Input testing data file, default = ./data/test_data.txt
-k KERNEL	SVM kernel,default=rbf
-c PENALTY_C	SVM penalty parameter C of the error term,default=1
-cv CV	SVM cross_validate,default=10

- 用-m 來指定分類器，dct=(Decision Tree),svm=(Support Vector Machine)
- Decision Tree:

  用 Training Data 訓練 Decision Tree，並將訓練出的 decisionTree 結果 output 至當前目錄的 tree.pdf 中。

- SVM:

  用 Training Data 訓練 Support Vector Machine，並將 cross_validate 的結果及 Testing 的 Accuracy、Precision、Recall 輸出。

- 需要安裝 graphviz:

```
$ apt-get install graphviz
```

Data Generator

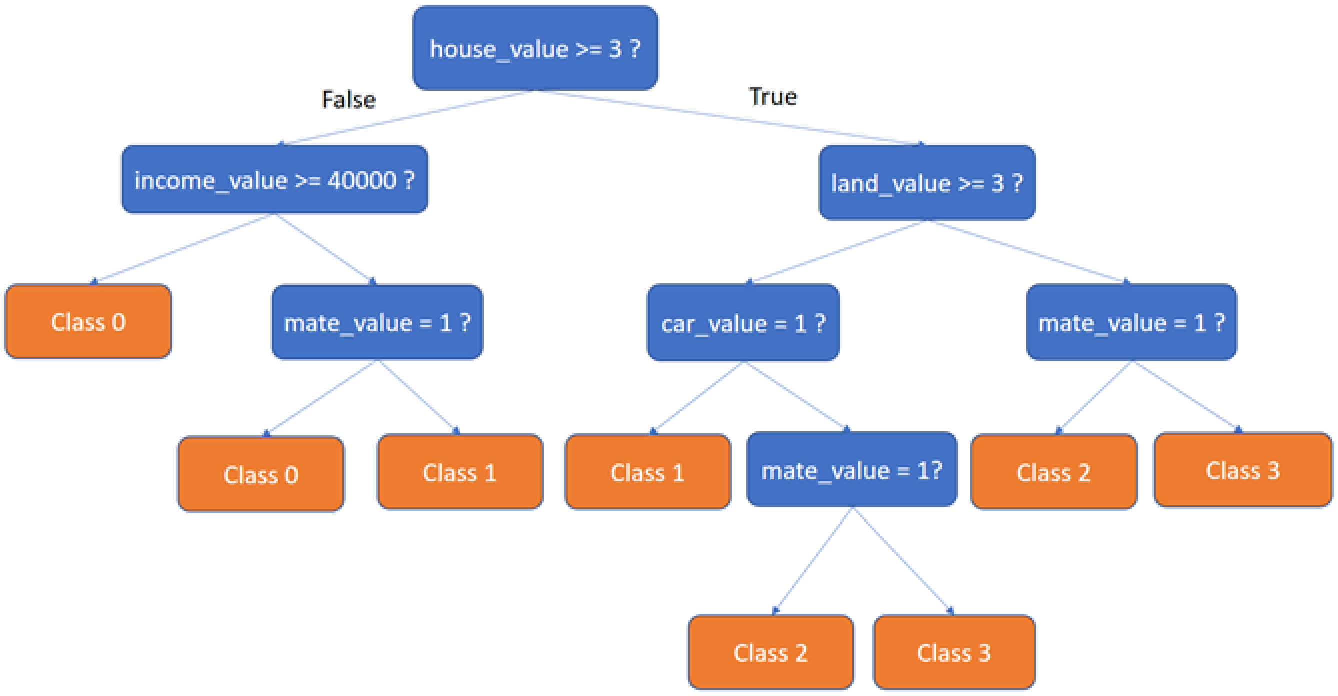
```
$ python3 data_generator.py [-h]
```

optional Options	**Description**
-h, --help	show this help message and exit
-train, TRAIN_SIZE	The number of training data you want to generate, default = 10000
–test, TEST_SIZE	The number of testing data you want to generate, default = 10000

- 執行後會在 data 資料夾內生成 10000 筆 training data(train_data.txt)及 testing data(test_data.txt)。

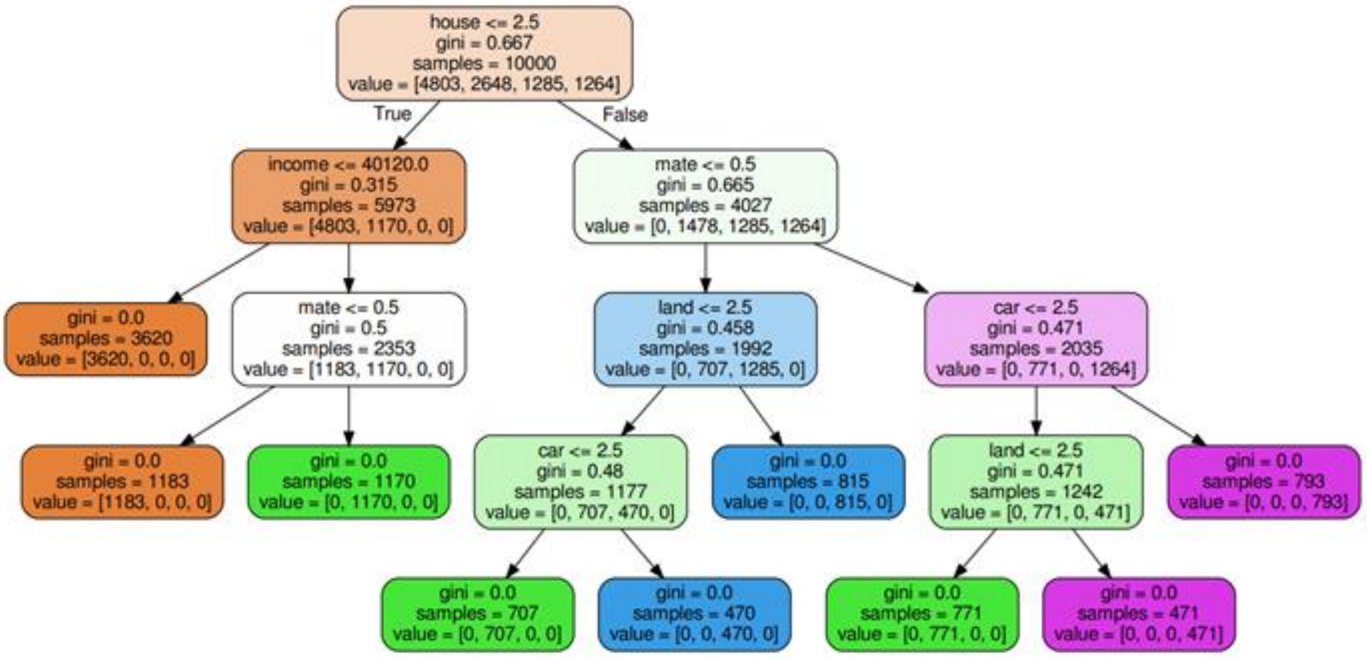
Absolutely Right Rules

- Attributes_list = house, car, land, income, mate, class
- house_value = [0, 5]
- car_value = [0, 5]
- land_value = [0, 5]
- income_value = [-50000, 100000]
- mate_value = [0, 1]
- class_vlaue = {0, 1, 2, 3}



Decision Tree

- Training size = 10000
- Testing size = 10000
- Criterion = Gini



Result metrics

- Accuracy = 0.9997
- Precision = 0.9999826689774697
- Recall = 0.9999677377726158

- 從結果圖可以得知，與 Absolutely Right Rules 相比，Decision tree 所建立出的 model 和實際的 rules 並非完全相同，但有很高的相似度，由於 Absolutely Right Rules 的規則很簡單，因此結果的 Accuracy, Precision, Recall 均有相當好的表現。

Support Vector Machine

- Penalty C: 1.0
- Cross Validate: 10

Kernel : rbf

Cross Validation Result:

CV No.	**Accuracy**	**Percision**	**Recall**	**Accuracy**	**Percision**	**Recall**
Type	Train	Train	Train	Test	Test	Test
cv0	0.9971	0.9979	0.9979	0.9940	0.9961	0.9953
cv1	0.9970	0.9978	0.9979	0.9980	0.9982	0.9990
cv2	0.9974	0.9980	0.9983	0.9940	0.9949	0.9965
cv3	0.9969	0.9977	0.9978	0.9970	0.9976	0.9980
cv4	0.9966	0.9974	0.9976	0.9970	0.9984	0.9972

CV No.	Accuracy	Percision	Recall	Accuracy	Percision	Recall
cv5	0.9961	0.9971	0.9973	0.9970	0.9976	0.9980
cv6	0.9968	0.9976	0.9977	0.9980	0.9990	0.9981
cv7	0.9974	0.9981	0.9982	0.9940	0.9953	0.9960
cv8	0.9970	0.9977	0.9979	0.9950	0.9966	0.9962
cv9	0.9972	0.9978	0.9981	0.9970	0.9980	0.9976
avg	0.9970	0.9977	0.9979	0.9961	0.9972	0.9972

Testing Result:

- Accuracy: 0.9969
- Precision: 0.9976627899295014
- Recall: 0.9977905324777975

Kernel: sigmoid

Cross Validation Result:

CV No.	Accuracy	Percision	Recall	Accuracy	Percision	Recall
Type	Train	Train	Train	Test	Test	Test
cv0	0.6478	0.6150	0.6020	0.6507	0.6169	0.6125
cv1	0.6443	0.6152	0.5984	0.6687	0.6406	0.6432

CV No.	Accuracy	Percision	Recall	Accuracy	Percision	Recall
cv2	0.6568	0.6282	0.6147	0.6687	0.6398	0.6319
cv3	0.6545	0.6308	0.6137	0.6194	0.5878	0.5671
cv4	0.6461	0.6201	0.5991	0.6670	0.6313	0.6152
cv5	0.6463	0.6214	0.6007	0.6396	0.6195	0.5945
cv6	0.6519	0.6267	0.6085	0.6176	0.5853	0.5553
cv7	0.6549	0.6289	0.6112	0.6767	0.6457	0.6284
cv8	0.6490	0.6207	0.6061	0.6423	0.6170	0.6011
cv9	0.6603	0.6299	0.6098	0.6513	0.6296	0.6088
avg	0.6512	0.6237	0.6064	0.6502	0.6214	0.6058

Testing Result:

- Accuracy: 0.6435
- Precision: 0.6204495628355513
- Recall: 0.6003172025937911

Kernel: poly

Cross Validation Result:

CV No.	Accuracy	Percision	Recall	Accuracy	Percision	Recall
Type	Train	Train	Train	Test	Test	Test
cv0	0.9617	0.9786	0.9653	0.9731	0.9850	0.9758
cv1	0.9631	0.9799	0.9663	0.9611	0.9776	0.9636
cv2	0.9615	0.9783	0.9651	0.9701	0.9826	0.9725
cv3	0.9627	0.9791	0.9662	0.9411	0.9694	0.9452
cv4	0.9617	0.9781	0.9653	0.9680	0.9844	0.9698
cv5	0.9627	0.9787	0.9661	0.9650	0.9805	0.9687
cv6	0.9637	0.9796	0.9670	0.9580	0.9785	0.9612
cv7	0.9597	0.9775	0.9633	0.9530	0.9733	0.9574
cv8	0.9619	0.9786	0.9653	0.9599	0.9743	0.9655
cv9	0.9622	0.9789	0.9656	0.9679	0.9799	0.9718
avg	0.9621	0.9787	0.9655	0.9617	0.9786	0.9652

Testing Result:

- Accuracy: 0.9607
- Precision: 0.978488520596513
- Recall: 0.9643157193348397

- 從結果可以看到，Support Vector Machine 在 Kernel 為 rbf 和 poly 時進行分類的效果也很不錯，而 sigmoid 則較差，但在簡單的規則下所定義出的 data，用 Decision Tree 這種

簡單的 model 反而效果還要比用 SVM 來得更好，因此要視問題來決定 model，而不是一味的使用特定的 model 來解決問題。