Chaithanya Chikkannaswamy

SUID : 387781563

Evolutionary Machine Learning – HW5

HW5: Use Genetic Programming for learning a tree model for the same problem. Compare with the previous results (HW1-4).

Dataset:

Link:
https://www.kaggle.com/chaithanya96/bankmarketing

The bank marketing dataset is the csv file with users' details used to predict the marketing decision. The decision yes or no is represented with binary numbers 1 and 0. The sample columns included are:

1. 'age',
2. 'job',
3. 'marital',
4. 'education',
5. 'default',
6. 'balance',
7. 'housing',
8. 'loan',
9. 'contact',
10. 'day',
11. 'month',
12. 'duration',
13. 'campaign',
14. 'pdays',
15. 'previous',
16. 'poutcome',
17. 'market?'

The screenshot of the sample dataset is attached below:

Chaithanya Chikkannaswamy

bank marketing 2

age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	market?
30	unemployed	married	primary	0	1787	0	0	cellular	19	oct	79	1	-1	0	unknown	0
33	services	married	secondary	0	4789	1	1	cellular	11	may	220	1	339	4	failure	0
35	management	single	tertiary	0	1350	1	0	cellular	16	apr	185	1	330	1	failure	0
30	management	married	tertiary	0	1476	1	1	unknown	3	jun	199	4	-1	0	unknown	0
59	blue-collar	married	secondary	0	0	1	0	unknown	5	may	226	1	-1	0	unknown	0
35	management	single	tertiary	0	747	0	0	cellular	23	feb	141	2	176	3	failure	0
36	self-employed	married	tertiary	0	307	1	0	cellular	14	may	341	1	330	2	other	0
39	technician	married	secondary	0	147	1	0	cellular	6	may	151	2	-1	0	unknown	0
41	entrepreneur	married	tertiary	0	221	1	0	unknown	14	may	57	2	-1	0	unknown	0
43	services	married	primary	0	-88	1	1	cellular	17	apr	313	1	147	2	failure	0
39	services	married	secondary	0	9374	1	0	unknown	20	may	273	1	-1	0	unknown	0
43	admin.	married	secondary	0	264	1	0	cellular	17	apr	113	2	-1	0	unknown	0
36	technician	married	tertiary	0	1109	0	0	cellular	13	aug	328	2	-1	0	unknown	0
20	student	single	secondary	0	502	0	0	cellular	30	apr	261	1	-1	0	unknown	1
31	blue-collar	married	secondary	0	360	1	1	cellular	29	jan	89	1	241	1	failure	0
40	management	married	tertiary	0	194	0	1	cellular	29	aug	189	2	-1	0	unknown	0
56	technician	married	secondary	0	4073	0	0	cellular	27	aug	239	5	-1	0	unknown	0
37	admin.	single	tertiary	0	2317	1	0	cellular	20	apr	114	1	152	2	failure	0
25	blue-collar	single	primary	0	-221	1	0	unknown	23	may	250	1	-1	0	unknown	0
31	services	married	secondary	0	132	0	0	cellular	7	jul	148	1	152	1	other	0
38	management	divorced	unknown	0	0	1	0	cellular	18	nov	96	2	-1	0	unknown	0
42	management	divorced	tertiary	0	16	0	0	cellular	19	nov	140	3	-1	0	unknown	0
44	services	single	secondary	0	106	0	0	unknown	12	jun	109	2	-1	0	unknown	0
44	entrepreneur	married	secondary	0	93	0	0	cellular	7	jul	125	2	-1	0	unknown	0
26	housemaid	married	tertiary	0	543	0	0	cellular	30	jan	169	3	-1	0	unknown	0

Code execution steps:

Genetic Programming

I have implemented the Genetic Programming to train a shallow feedforward neural network for a 2-class classification task using the deap library available in python. The accuracy is then compared with the accuracy calculated using Genetic Algorithm, CMA Evolutionary Strategy, particle swarm optimization and LCS implemented in previous HWs.

1. Deap python library is installed to implement the Genetic Programming approach to train the data.
2. Pandas data frame available in python is used to read the data set.
3. The data is split into training and testing data for the further implementation by using train_test_split() and specifying the test_size as 0.2.
4. Once the training and testind data are ready, Genetic Programming is implemented on the dataset by specifying the parameters.
5. PrimitiveSetTyped() is used to add the different operators for float and Boolean data.
6. FitnessMAx and Individuals are determined using the base.Fitness and Tree representation of data i.e. gp.PrimitiveTree.
7. The prediction accuracy is calculated for the model based.

Chaithanya Chikkannaswamy

The snapshot of data set after converting it to discreet data:

Out[121]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	market?
0	0	unemployed	married	primary	0	0	0	0	cellular	1	oct	0	1	-1	0	unknown	0
1	0	services	married	secondary	0	0	1	1	cellular	0	may	0	1	339	4	failure	0
2	0	management	single	tertiary	0	0	1	0	cellular	1	apr	0	1	330	1	failure	0
3	0	management	married	tertiary	0	0	1	1	unknown	0	jun	0	4	-1	0	unknown	0
4	1	blue-collar	married	secondary	0	0	1	0	unknown	0	may	0	1	-1	0	unknown	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4516	0	services	married	secondary	0	0	1	0	cellular	2	jul	0	5	-1	0	unknown	0
4517	1	self-employed	married	tertiary	1	0	1	1	unknown	0	may	0	1	-1	0	unknown	0
4518	1	technician	married	secondary	0	0	0	0	cellular	1	aug	0	11	-1	0	unknown	0
4519	0	blue-collar	married	secondary	0	0	0	0	cellular	0	feb	0	4	211	3	other	0
4520	0	entrepreneur	single	tertiary	0	0	1	1	cellular	0	apr	0	2	249	7	other	0

4521 rows × 17 columns

The snapshot of data set after dropping the non-usable data and converting the required column for prediction to bool data type:

```
print(df.columns)
df[10] = df[10].astype('int').astype('bool')
print(df.head(10))
```

```
Int64Index([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10], dtype='int64')
    0  1     2  3  4   5    6  7    8  9      10
1   30  0  1787  0  0  19   79  1   -1  0  False
2   33  0  4789  1  1  11  220  1  339  4  False
3   35  0  1350  1  0  16  185  1  330  1  False
4   30  0  1476  1  1   3  199  4   -1  0  False
5   59  0     0  1  0   5  226  1   -1  0  False
6   35  0   747  0  0  23  141  2  176  3  False
7   36  0   307  1  0  14  341  1  330  2  False
8   39  0   147  1  0   6  151  2   -1  0  False
9   41  0   221  1  0  14   57  2   -1  0  False
10  43  0   -88  1  1  17  313  1  147  2  False
```

+ Code + Markdown

The snapshot of splitting the data set into train and test data:

```
train, test = train_test_split(df, test_size=0.2)
print(train.shape)
print(train.head(10))
train_data = train.values.tolist()
print(test.head(10))
```

```
(3616, 11)
      0  1     2  3  4   5    6  7   8  9      10
4256  42  0  1331  0  0  23  698  2  -1  0  False
4489  31  0   315  0  0  30  130  1   2  1  False
3895  38  0     0  1  0  17  213  2  -1  0  False
2834  38  0    62  0  0  19  212  1  -1  0  False
860   51  0     4  1  1  20   74  5  -1  0  False
1890  42  0  1205  0  0   8  376  2  -1  0   True
3909  36  0   167  1  0  28   57  1  -1  0  False
841   28  1  -298  1  0   3  559  7  -1  0  False
3664  37  0   196  1  0   5   66  3  -1  0  False
1404  29  0   912  1  0  13  785  1  -1  0  False
      0  1     2  3  4   5    6   7    8  9      10
509   41  0   428  1  0  12   92   1   -1  0  False
1689  38  0  -278  1  0  28  143   2   -1  0  False
3856  47  0    -9  0  0  14  457   2   -1  0  False
1636  33  0   687  1  0  14   72   2  370  1  False
628   57  0   374  1  1  16  236   1   -1  0  False
4468  43  0  1577  1  0  19   87   1   -1  0  False
84    52  0   657  0  0   7  398   2  460  2   True
3533  42  0  3620  1  0  27   22  16   -1  0  False
2731  52  0     9  0  1   6   44   1   -1  0  False
3337  37  0   215  1  0   6   61   1   -1  0  False
```

The snapshot of testing and training results of the data:

```
train_res = train[10]
test_res = test[10]
print(train_res)
```

```
4256    False
4489    False
3895    False
2834    False
860     False
        ...
2284    False
3116    False
2272    False
2221     True
716     False
Name: 10, Length: 3616, dtype: bool
```

The snapshot of the accuracy rate:

```
[72]: accuracy_score(test_res, predict(pop[0]))
```

```
Out[72] 0.7933701657458564
```

Chaithanya Chikkannaswamy

Conclusion:

1. The **Genetic Programming** resulted in **79%** accuracy.
2. The **Learning Classifier System** resulted in the accuracy of **99%, 88%, 99% and 99%** based on the columns.
3. The **Particle Swarm Optimization** resulted in the accuracy of 0.88 i.e. **88**%.
4. The **CMA-ES** resulted in 0.715 test accuracy.
5. The **Genetic Neural Network** resulted in a test accuracy of 0.89 and the **Sequential Neural Network** resulted in a test accuracy of 0.88.

References

1. https://github.com/trevorstephens/gplearn/blob/master/gplearn/fitness.py
2. https://github.com/sighmin/gpstocks
3. https://docs.google.com/file/d/0B4JHGiC-rWKmbm9MYWx2b1VNV2c/edit
4. https://docs.google.com/file/d/0B4JHGiC-rWKmWThhbWtwVngxbnM/edit
5. Discussed with classmates.