

CS773-HOMEWORK-1

Step 1: Create an ARFF file `bean_training_data.arff` with the same 9 fields.

Solution

(a) Since the input file is in tab delimited, we need to first convert it into CSV so that Weka can read it. Open `bean_training_data.txt` with Microsoft Excel or R and convert the file into .CSV.

(b) Read the `bean_training_data.csv` into Weka and convert it into `bean_training_data.arff` as shown below. We can also change the field names while exporting to Weka in ArffViewer.

Screenshots

Text Import Wizard - Step 2 of 3

This screen lets you set the delimiters your data contains. You can see how your text is affected in the preview below.

Delimiters

☒ Tab
☐ Semicolon
☐ Comma
☐ Space
☐ Other:

☐ Treat consecutive delimiters as one

Text qualifier:

Data preview

Area	Perimeter	Major_axis_length	Minor_axis_length	Aspect_ratio	Eccentrici
28395	610	208	174	1.2	0.55
28734	638	201	183	1.1	0.41
29380	624	213	176	1.21	0.56
30008	646	211	183	1.15	0.5

Cancel < Back Next > Finish

Input CSV file

A,P,L,l,K,Ec,C,Ed,Class
28395,610,208,174,1.2,0.55,28715,190,SEKER
28734,638,201,183,1.1,0.41,29172,191,SEKER
29380,624,213,176,1.21,0.56,29690,193,SEKER
30008,646,211,183,1.15,0.5,30724,195,SEKER
30140,620,202,190,1.06,0.33,30417,196,SEKER
30279,635,213,182,1.17,0.52,30600,196,SEKER
30477,670,211,184,1.15,0.49,30970,197,SEKER
30519,630,213,183,1.17,0.51,30847,197,SEKER
30685,636,214,183,1.17,0.51,31044,198,SEKER
30834,632,217,181,1.2,0.55,31120,198,SEKER
30917,641,214,184,1.16,0.5,31280,198,SEKER
31091,639,210,188,1.12,0.45,31458,199,SEKER
31107,641,215,185,1.16,0.51,31423,199,SEKER
31158,643,216,184,1.18,0.53,31492,199,SEKER
31158,641,212,187,1.13,0.47,31474,199,SEKER
31178,637,213,187,1.14,0.48,31520,199,SEKER
31202,644,216,184,1.17,0.52,31573,199,SEKER
31203,640,215,185,1.16,0.51,31558,199,SEKER
31272.639.212.188.1.13.0.47.31593.200.SEKER

Output Arff File

```

@relation bean_training_data_file

@attribute Area numeric
@attribute Perimeter numeric
@attribute 'Major axis length' numeric
@attribute 'Minor axis length' numeric
@attribute 'Aspect ratio' numeric
@attribute Eccentricity numeric
@attribute 'Convex area' numeric
@attribute 'Equivalent diameter' numeric
@attribute Class {SEKER, BARBUNYA, BOMBAY, CALI, HOROZ, SIRA, DERMASON}



@data
20420,525,184,142,1.29,0.63,20684,161,DERMASON
20464,528,191,136,1.4,0.7,20772,161,DERMASON
20548,525,184,143,1.29,0.63,20825,162,DERMASON
20711,525,186,142,1.31,0.65,20988,162,DERMASON
20786,535,201,132,1.52,0.75,21057,163,DERMASON
20942,531,191,140,1.37,0.68,21191,163,DERMASON
21101,534,185,146,1.27,0.62,21462,164,DERMASON
21314,537,194,140,1.38,0.69,21587,165,DERMASON
21348,531,192,142,1.35,0.67,21590,165,DERMASON
21397,535,193,142,1.36,0.68,21731,165,DERMASON
21405,535,191,143,1.34,0.66,21737,165,DERMASON
21479,544,199,138,1.44,0.72,21762,165,DERMASON
21558,539,197,139,1.41,0.71,21808,166,DERMASON
21570,538,197,140,1.41,0.7,21820,166,DERMASON

```

Step 2. Create a corresponding csv file bean_training_data.csv

Solution

Either we can use Excel or R to convert tab delimited to .CSV file or we can use previously created bean_training_data.arff file into CSV using Arffviewer in Weka.

 bean_training_data_file	5/15/2024 12:49 AM	ARFF Data File	611 KB
 bean_training_data_file_from_arff	5/15/2024 12:54 AM	Microsoft Excel Comma Separated Values File	611 KB

File Edit View

```
Area,Perimeter,'Major axis length','Minor axis length','Aspect ratio',Eccentricity,'Convex area','Equivalent diameter',Class
20420,525,184,142,1.29,0.63,20684,161,DERMASON
20464,528,191,136,1.4,0.7,20772,161,DERMASON
20548,525,184,143,1.29,0.63,20825,162,DERMASON
20711,525,186,142,1.31,0.65,20988,162,DERMASON
20786,535,201,132,1.52,0.75,21057,163,DERMASON
20942,531,191,140,1.37,0.68,21191,163,DERMASON
21101,534,185,146,1.27,0.62,21462,164,DERMASON
21314,537,194,140,1.38,0.69,21587,165,DERMASON
21348,531,192,142,1.35,0.67,21590,165,DERMASON
21397,535,193,142,1.36,0.68,21731,165,DERMASON
21405,535,191,143,1.34,0.66,21737,165,DERMASON
21479,544,199,138,1.44,0.72,21762,165,DERMASON
21558,539,197,139,1.41,0.71,21808,166,DERMASON
21570,538,197,140,1.41,0.7,21820,166,DERMASON
```

Step 3. Using Weka, with Output set to Plain text (under more options), runbean_training_data.csv file using (i) Linear Regression (ii) MultilayerPerceptron classifiers(available under function under classify). Show the predicted bean class for each of the 9 records in the test file (bean_test_file.txt or related bean_test_file.csv) along with the actual class shown

in the test file. You need to convert .txt to the corresponding formats.

Solution

(a).Linear Regression

Classifier

Choose **LinearRegression** -S 0 -R 1.0E-8 -num-decimal-places 4

Test options

☒ Use training set
☐ Supplied test set Set...
☐ Cross-validation Folds 10
☐ Percentage split % 66

More options...

(Num) Class

Start

Stop

Result list (right-click for options)

23:53:31 - functions.LinearRegression

Classifier output

```

=== Run information ===

Scheme:      weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4
Relation:    bean_training_data_file
Instances:   13611
Attributes:  9
              Area
              Perimeter
              Major axis length
              Minor axis length
              Aspect ratio
              Eccentricity
              Convex area
              Equivalent diameter
              Class

Test mode:   evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Class =

      0      * Area +
    -0.0152 * Major axis length +
     5.3115 * Aspect ratio +
    -10.2844 * Eccentricity +
       8.2024

Time taken to build model: 0.16 seconds

=== Predictions on training set ===

inst#   actual   predicted   error
   1      6      6.031      0.031
   2      6      7.049      1.049
   3      6      5.915     -0.085
   4      6      6.25       0.25
   5      6      7.658      1.658
   6      6      6.122      0.122
   7      6      6.357      0.357
   8      6      6.228      0.228
   9      6      6.214      0.214
  10      6      5.918     -0.082
  11      6      6.266      0.266
  12      6      6.63       0.63

=== Evaluation on training set ===

Time taken to test model on training data: 2.54 seconds

=== Summary ===

Correlation coefficient      0.5212
Mean absolute error          1.2758
Root mean squared error      1.5606
Relative absolute error      83.6524 %
Root relative squared error   85.3448 %
Total Number of Instances    13611

```

(b). MultilayerPerceptron

Classifier: **MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a**

Test options

- ☒ Use training set
- ☐ Supplied test set
- ☐ Cross-validation Folds: 10
- ☐ Percentage split %: 66

More options...

(Nom) Class

Start Stop

Result list (right-click for options)

- 01:24:58 - functions.Logistic
- 01:26:37 - functions.Logistic
- 01:27:55 - functions.Logistic
- 01:29:59 - functions.MultilayerPerceptron

Classifier output

Attrib Area 3.7962670140475193
 Attrib Perimeter 0.01937349591782552
 Attrib Major axis length 2.576836129339923
 Attrib Minor axis length 4.735766809484082
 Attrib Aspect ratio -3.305051216974391
 Attrib Eccentricity -2.6497904903634657
 Attrib Convex area 3.365884858427008
 Attrib Equivalent diameter 3.8036805465325085

Class DERMASON
 Input
 Node 0

Class SEKER
 Input
 Node 1

Class SIRA
 Input
 Node 2

Class HOROZ
 Input
 Node 3

Class BARBUNYA
 Input
 Node 4

Class CALI
 Input
 Node 5

Class BOMBAY
 Input
 Node 6

Time taken to build model: 7.8 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.05 seconds

=== Summary ===

Correctly Classified Instances	12433	91.3452 %
Incorrectly Classified Instances	1178	8.6548 %
Kappa statistic	0.8952	
Mean absolute error	0.0387	
Root mean squared error	0.1369	
Relative absolute error	16.3671 %	
Root relative squared error	39.8167 %	
Total Number of Instances	13611	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.935	0.039	0.893	0.935	0.913	0.882	0.991	0.972	DERMASON
	0.935	0.010	0.942	0.935	0.939	0.928	0.992	0.959	SEKER
	0.836	0.032	0.864	0.836	0.850	0.815	0.980	0.917	SIRA
	0.947	0.009	0.944	0.947	0.945	0.936	0.994	0.978	HOROZ
	0.892	0.008	0.920	0.892	0.906	0.896	0.990	0.961	BARBUNYA
	0.915	0.009	0.934	0.915	0.924	0.914	0.994	0.964	CALI
	1.000	0.000	0.998	1.000	0.999	0.999	1.000	1.000	BOMBAY
Weighted Avg.	0.913	0.021	0.913	0.913	0.913	0.893	0.990	0.959	

=== Confusion Matrix ===

	a	b	c	d	e	f	g	<-- classified as
3314	40	185	6	1	0	0	0	a = DERMASON
60	1896	60	0	11	0	0	0	b = SEKER
316	62	2204	40	13	1	0	0	c = SIRA
21	0	56	1826	12	13	0	0	d = HOROZ
0	12	27	11	1179	92	1	0	e = BARBUNYA
0	3	18	52	65	1492	0	0	f = CALI
0	0	0	0	0	0	522	0	g = BOMBAY

Convert bean_test_file.txt to bean_test_file.csv either using Excel or R

We need to save model for making prediction in Weka, save log.model and mlp.model for LogisticRegression and MLP.

(a) Prediction with Linear Regression

Linear Regression Model

Class =

-0 * Convex area +
6.4401

Time taken to build model: 0 seconds

=== Predictions on test set ===

inst#	actual	predicted	error
1	6	5.28	-0.72
2	4	5.448	1.448
3	6	5.251	-0.749
4	5	4.308	-0.692
5	1	3.945	2.945
6	3	3.329	0.329
7	2	1.476	-0.524
8	7	4.922	-2.078
9	1	3.935	2.935
10	2	1.445	-0.555
11	4	5.448	1.448
12	7	4.928	-2.072
13	3	3.338	0.338
14	5	4.273	-0.727
15	5	4.303	-0.697
16	7	4.922	-2.078
17	4	5.449	1.449

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

(b)MLP Prediction

Time taken to build model: 0.02 seconds

=== Predictions on test set ===

inst#	actual	predicted	error	prediction
1	1:SEKER	1:SEKER	0.863	
2	2:DERMASON	2:DERMASON	0.848	
3	1:SEKER	1:SEKER	0.9	
4	3:HOROZ	3:HOROZ	0.903	
5	4:BARBUNYA	4:BARBUNYA	0.804	
6	5:CALI	5:CALI	0.846	
7	6:BOMBAY	6:BOMBAY	0.889	
8	7:SIRA	7:SIRA	0.854	
9	4:BARBUNYA	4:BARBUNYA	0.804	
10	6:BOMBAY	6:BOMBAY	0.882	
11	2:DERMASON	2:DERMASON	0.865	
12	7:SIRA	7:SIRA	0.86	
13	5:CALI	5:CALI	0.857	
14	3:HOROZ	3:HOROZ	0.894	
15	3:HOROZ	3:HOROZ	0.884	
16	7:SIRA	7:SIRA	0.861	
17	2:DERMASON	2:DERMASON	0.866	

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

Step 4. Repeat step 3 with bean_training_data.arff file. You should get exactly the same results as in Step 3

(a) Prediction with Linear Regression

Linear Regression Model

Class =

-0 * Convex area +
6.4401

Time taken to build model: 0 seconds

=== Predictions on test set ===

inst#	actual	predicted	error
1	6	5.28	-0.72
2	4	5.448	1.448
3	6	5.251	-0.749
4	5	4.308	-0.692
5	1	3.945	2.945
6	3	3.329	0.329
7	2	1.476	-0.524
8	7	4.922	-2.078
9	1	3.935	2.935
10	2	1.445	-0.555
11	4	5.448	1.448
12	7	4.928	-2.072
13	3	3.338	0.338
14	5	4.273	-0.727
15	5	4.303	-0.697
16	7	4.922	-2.078
17	4	5.449	1.449

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

(b)MLP Prediction

```
Time taken to build model: 0.02 seconds
```

```
=== Predictions on test set ===
```

inst#	actual	predicted	error	prediction
1	1:SEKER	1:SEKER	0.863	
2	2:DERMASON	2:DERMASON	0.848	
3	1:SEKER	1:SEKER	0.9	
4	3:HOROZ	3:HOROZ	0.903	
5	4:BARBUNYA	4:BARBUNYA	0.804	
6	5:CALI	5:CALI	0.846	
7	6:BOMBAY	6:BOMBAY	0.889	
8	7:SIRA	7:SIRA	0.854	
9	4:BARBUNYA	4:BARBUNYA	0.804	
10	6:BOMBAY	6:BOMBAY	0.882	
11	2:DERMASON	2:DERMASON	0.865	
12	7:SIRA	7:SIRA	0.86	
13	5:CALI	5:CALI	0.857	
14	3:HOROZ	3:HOROZ	0.894	
15	3:HOROZ	3:HOROZ	0.884	
16	7:SIRA	7:SIRA	0.861	
17	2:DERMASON	2:DERMASON	0.866	

```
=== Evaluation on test set ===
```

```
Time taken to test model on supplied test set: 0.01 seconds
```

Step 5. Consider the raisin data

(<https://archive.ics.uci.edu/ml/datasets/Raisin+Dataset>) predicting the type of raisins. You are provided the training set (Raisin_data_set) and the test set (raisin_test_data) for this data. By applying “Linear Regression” and “Gaussian Processes” options, predict the output for the test cases. Since output class is categorical, for linear regression, convert the output class to a numeric by mapping each class to an integer 1-7. Similarly, map the regression output back to an actual class based on nearest integer value.

Solution: Since the training and test datasets are in .xls format we need to convert them first to .csv

Csv Training data

File	Edit	View
Area,MajorAxisLength,MinorAxisLength,Eccentricity,ConvexArea,Extent,Perimeter,Class		
87524,442.2460114,253.291155,0.819738392,90546,0.758650579,1184.04,2		
75166,406.690687,243.0324363,0.801805234,78789,0.68412957,1121.786,2		
90856,442.2670483,266.3283177,0.798353619,93717,0.637612812,1208.575,2		
45928,286.5405586,208.7600423,0.684989217,47336,0.699599385,844.162,2		
79408,352.1907699,290.8275329,0.56401133,81463,0.792771926,1073.251,2		
49242,318.125407,200.12212,0.777351277,51368,0.658456354,881.836,2		
42492,310.1460715,176.1314494,0.823098681,43904,0.665893562,823.796,2		
60952,332.4554716,235.429835,0.706057518,62329,0.74359819,933.366,2		
42256,323.1896072,172.5759261,0.845498789,44743,0.698030924,849.728,2		
64380,366.9648423,227.7716147,0.784055626,66125,0.664375716,981.544,2		
80437,449.4545811,232.3255064,0.856042518,84460,0.674235757,1176.305,2		
43725,301.3222176,186.9506295,0.784258452,45021,0.697068248,818.873,2		
43441,276.6108288,201.8131355,0.683882337,45133,0.690855598,803.748,2		
76792,338.8575454,291.3592017,0.510583813,78842,0.772322237,1042.77,2		
74167,387.7989307,247.8581228,0.769089738,76807,0.680181585,1084.729,2		
33565,261.5543311,167.7084908,0.767374275,35794,0.68155052,751.413,2		
64670,403.0839752,206.4846437,0.858829168,66419,0.75677257,1028.445,2		
64762,354.2939396,235.7524629,0.746473726,66713,0.694998015,981.509,2		
43295,304.2844667,182.8110368,0.799406959,44714,0.713838189,814.68,2		
70699,418.6985723,216.5960537,0.855799392,72363,0.728075054,1061.321,2		
69776 354 1769124 252 529208 0 701160962 71849 0 734398534 1035 501 2		

Csv Test Data

File	Edit	View
Area,MajorAxisLength,MinorAxisLength,Eccentricity,ConvexArea,Extent,Perimeter,Class		
94211,450,269,0.80,96340,0.72,1195,Besni		
87524,442,253,0.82,90546,0.76,1184,Kecimen		
75166,407,243,0.80,78789,0.68,1122,Kecimen		
86852,456,249,0.84,90550,0.61,1208,Besni		
91464,433,273,0.78,93852,0.72,1182,Besni		
93441,397,301,0.65,95370,0.72,1158,Besni		
90856,442,266,0.80,93717,0.64,1209,Kecimen		
45928,287,209,0.68,47336,0.70,844,Kecimen		

Linear Regression on Train Data

Classifier

Choose **LinearRegression** -S 0 -R 1.0E-8 -num-decimal-places 4

Test options

☒ Use training set
☐ Supplied test set Set...
☐ Cross-validation Folds
☐ Percentage split %

More options...

(Num) Class

Start Stop

Result list (right-click for options)

22:53:15 - functions.LinearRegression

Classifier output

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: Raisin_training_data_converted

Instances: 900

Attributes: 8

Area

MajorAxisLength

MinorAxisLength

Eccentricity

ConvexArea

Extent

Perimeter

Class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Class =

0 * Area +

-0.0013 * MajorAxisLength +

-0.0048 * MinorAxisLength +

-1.6611 * Eccentricity +

-0.0006 * Perimeter +

4.6912

Time taken to build model: 0.04 seconds

=== Predictions on training set ===

inst#	actual	predicted	error
1	2	1.415	-0.585
2	2	1.499	-0.501
3	2	1.396	-0.604
4	2	1.988	-0.012
5	2	1.791	-0.209
6	2	1.834	-0.166
7	2	1.873	-0.127
8	2	1.808	-0.192
9	2	1.82	-0.18
10	2	1.664	-0.336
11	2	1.406	-0.594
12	2	1.908	-0.092
13	2	2.043	0.043
14	2	1.895	-0.105
15	2	1.569	-0.431

Gaussian Processes on Train Data

Choose

GaussianProcesses -L 1.0 -N 0 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -S 1

Test options

☒ Use training set

☐ Supplied test set

Set...

☐ Cross-validation

Folds 10

☐ Percentage split

% 66

More options...

(Num) Class

Start Stop

Result list (right-click for options)

22:53:15 - functions.LinearRegression
 22:55:02 - functions.GaussianProcesses

Classifier output

```

=== Run information ===

Scheme:      weka.classifiers.functions.GaussianProcesses -L 1.0 -N 0 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -S 1
Relation:    Raisin_training_data_converted
Instances:   900
Attributes:  8
              Area
              MajorAxisLength
              MinorAxisLength
              Eccentricity
              ConvexArea
              Extent
              Perimeter
              Class

Test mode:   evaluate on training data

=== Classifier model (full training set) ===

Gaussian Processes

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

All values shown based on: Normalize training data

Average Target Value : 0.5
Inverted Covariance Matrix:
  Lowest Value = -0.060574422975717766
  Highest Value = 0.9986999764294886
Inverted Covariance Matrix * Target-value Vector:
  Lowest Value = -1.007157688298649
  Highest Value = 1.8662039735907896

Time taken to build model: 0.68 seconds

=== Predictions on training set ===

inst#   actual   predicted   error
1       2       1.568      -0.432
2       2       1.542      -0.458
3       2       1.365      -0.635
4       2       1.847      -0.153
5       2       1.783      -0.217
6       2       1.726      -0.274
7       2       1.774      -0.226
8       2       1.803      -0.197
9       2       1.804      -0.196
10      2       1.616      -0.384
11      2       1.446      -0.554
12      2       1.833      -0.167
          
```

Prediction with Linear Regression

inst#	actual	predicted	round-off	r	actual_class	predicted_class
1	1	1.366	1		Besni	Besni
2	2	1.559	2		Kecimen	Kecimen
3	2	1.68	2		Kecimen	Kecimen
4	1	1.608	2		Besni	Kecimen
5	1	1.317	1		Besni	Besni
6	1	0.977	1		Besni	Besni
7	1	1.402	1		Besni	Besni

8	2	2.092	2	Kecimen	Kecimen
---	---	-------	---	---------	---------

Prediction with GaussianProcess

inst#	actual	predicted	roundoff	actual_class	predicted_class
1	1	1.437	1	Besni	Besni
2	2	1.582	2	Kecimen	Kecimen
3	2	1.54	1	Kecimen	Besni
4	1	1.418	1	Besni	Besni
5	1	1.42	1	Besni	Besni
6	1	1.191	1	Besni	Besni
7	2	1.355	1	Kecimen	Besni
8	2	1.658	2	Kecimen	Kecimen