

Classification Algorithms Report

1. Objective

The objective of this assignment is to learn how to select parameters for classification algorithms, evaluate the performance of trained models, and preprocess data. Four classification algorithms are evaluated on the E. coli dataset: K-Nearest Neighbors, Decision Tree, Random Forest, and Support Vector Machines.

Weighting: distance

Performance:

Accuracy: 87.1%

Precision: (0.93, 0.73, 1.0, 1.0, 0.625, 1.0, 1.0, 1.0)

Recall: (1.0, 0.73, 0.79, 1.0, 0.625, 0.5, 1.0, 1.0)

F1-score: (0.96, 0.73, 0.79, 1.0, 0.625, 0.67, 1.0, 1.0)

Decision Tree

Spyder (Python 3.12)

```
7 import sklearn
8 import sklearn.model_selection
9 import sklearn.neighbors
10 import sklearn.metrics
11 from sklearn.model_selection import train_test_split
12 from sklearn import tree
13 from sklearn import ensemble
14 from sklearn import svm
15 import pandas as pd
16
17 # Load dataset
18 df = pd.read_csv('D:/AI/39_Ecoli/ecoli.data', sep='\\s+', header=None)
19
20 X = df.iloc[:, 1:-1] # Skips the first column (ID) and selects features
21 y = df.iloc[:, -1] # Last column as target
22
23 # Split into training and test sets
24 X_train, X_test, y_train, y_test = train_test_split(
25     X, y, test_size=0.3, random_state=42, stratify=y
26 )
27
28 # Feature scaling
29 scaler = sklearn.preprocessing.StandardScaler()
30 X_train = scaler.fit_transform(X_train)
31 X_test = scaler.transform(X_test)
32
33 # Initialize classifier
34 # k=300 = sklearn.neighbors.NeighborsClassifier(n_neighbors=3, weights='
35 # k=10 = tree.DecisionTreeClassifier(criterion='gini', max_depth=3, min_
36 # k=10 = ensemble.RandomForestClassifier(criterion='entropy')
37 # k=10 = svm.SVC()
38 # k=10 = svm.SVC()
39
40 # Predict on the test set
41 y_pred = klasif.predict(X_test)
42
43 # Evaluate
44 accuracy = sklearn.metrics.accuracy_score(y_test, y_pred)
```

Variable Explorer

Name	Type	Size	Value
accuracy	float	1	0.7623762376237624
conf_mat	Array of int64	(8, 8)	[[43 0 0 ... 0 0 0] [1 17 0 ... 0 0 0]
df	DataFrame	(336, 9)	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8
f1	Array of float64	(8,)	[0.95555556 0.70833333 0. 0. 0.45454545 0.5 0. ...]
klasif	tree._classes.DecisionTreeClassifier	1	DecisionTreeClassifier object of sklearn.tree._classes module
precision	Array of float64	(8,)	[0.91489362 0.68 1. 1. 0.41666667 1. 0. ...]
recall	Array of float64	(8,)	[1. 0.73913043 0. 0. 0.5 0.33333333 0. ...]
scaler	preprocessing._data.StandardScaler	1	StandardScaler object of sklearn.preprocessing._data module
X	DataFrame	(336, 7)	Column names: 1, 2, 3, 4, 5, 6, 7

Console

```
In [11]: runfile('D:/AI/assignment1.py', wdir='D:/AI')
Accuracy on test set: 0.762376237624
Precision on test set: [0.91489362 0.68 1. 1. 0.41666667 1.
0. 0.90909091]
Recall on test set: [1. 0.73913043 0. 0. 0.5 0.33333333
0. 0.625]
F1 on test set: [0.95555556 0.70833333 0. 0. 0.45454545 0.5
0. 0.74074074]
[[43 0 0 0 0 0 0 0]
[ 1 17 0 0 0 0 0 0]
[ 0 0 0 0 0 1 0]
[ 0 1 0 0 0 0 0]
[ 0 5 0 0 0 0 0]
[ 1 0 0 0 1 2 1]
[ 1 0 0 0 0 0 0]
[ 1 2 0 0 1 0 2 10]]

In [12]:
```

Python Console History

conda: base (Python 3.12.7) | Completions: conda | LSP: Python | Line 37, Col 1 | UTF-8 | CRLF | RW | Mem 81%

208 PM 3/17/2025

Variable Explorer

Name	Type	Size	Value
accuracy	float	1	0.7623762376237624
conf_mat	Array of int64	(8, 8)	[[43 0 0 ... 0 0 0] [1 17 0 ... 0 0 0]
df	DataFrame	(336, 9)	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8
f1	Array of float64	(8,)	[0.95555556 0.70833333 0. 0. 0.45454545 0.5 0. ...]
klasif	tree._classes.DecisionTreeClassifier	1	DecisionTreeClassifier object of sklearn.tree._classes module
precision	Array of float64	(8,)	[0.91489362 0.68 1. 1. 0.41666667 1. 0. ...]
recall	Array of float64	(8,)	[1. 0.73913043 0. 0. 0.5 0.33333333 0. ...]
scaler	preprocessing._data.StandardScaler	1	StandardScaler object of sklearn.preprocessing._data module
X	DataFrame	(336, 7)	Column names: 1, 2, 3, 4, 5, 6, 7
X_test	Array of float64	(101, 7)	[[0.69674572 1.23720912 -0.1877293 ... 1.26529552 -0.4209401 -1. ...]
X_train	Array of float64	(235, 7)	[[-1.4084969 0.50836099 -0.1877293 ... -1.05705999 0.14217711 0 ...]
y	Series	(336,)	Series object of pandas.core.series module
y_pred	Array of object	(101,)	ndarray object of numpy module
y_test	Series	(101,)	Series object of pandas.core.series module
y_train	Series	(235,)	Series object of pandas.core.series module

Help Variable Explorer Plots Files

Parameters:

Criterion: Gini impurity

Maximum Depth: 3

Minimum Samples Split: 5

Performance:

Accuracy: 76.2%

Precision: (0.91, 0.68, 1.0, 1.0, 0.42, 1.0, 1.0, 1.0)

Recall: (1.0, 0.74, 0.79, 1.0, 0.5, 0.33, 0.0, 1.0)

F1-score: (0.96, 0.71, 0.79, 1.0, 0.45, 0.5, 0.0, 1.0)

Random Forest

Spyder (Python 3.12)

File Edit Search Source Run Debug Consoles Projects Tools View Help

D:\AI\assignment1.py

```
4 @author: gintaustad
5 """
6
7
8 import sklearn
9 import sklearn.model_selection
10 import sklearn.neighbors
11 from sklearn.model_selection import train_test_split
12 from sklearn import tree
13 from sklearn import ensemble
14 from sklearn import svm
15 import pandas as pd
16
17 # Load dataset
18 df = pd.read_csv('D:/AI/39_Ecoli/ecoli.data', sep='\\s+', header=None)
19
20 X = df.iloc[:, 1:-1] # Skips the first column (ID) and selects features
21 y = df.iloc[:, -1] # Last column as target
22
23 # Split into training and test sets
24 X_train, X_test, y_train, y_test = train_test_split(
25     X, y, test_size=0.3, random_state=42, stratify=y
26 )
27
28 # Feature scaling
29 scaler = sklearn.preprocessing.StandardScaler()
30 X_train = scaler.fit_transform(X_train)
31 X_test = scaler.transform(X_test)
32
33 # Initialize classifier
34 #k1 = sklearn.neighbors.KNeighborsClassifier(n_neighbors=3, weights='distance')
35 #k2 = tree.DecisionTreeClassifier(criterion='gini', max_depth=3, min_samples_split=10)
36 k3 = ensemble.RandomForestClassifier(criterion='entropy')
37 k4 = svm.SVC()
38 k5 = k3
39 k6 = k4
40 k7 = k5
41 k8 = k6
42 k9 = k7
43 k10 = k8
44 k11 = k9
45 k12 = k10
46 k13 = k11
47 k14 = k12
48 k15 = k13
49 k16 = k14
50 k17 = k15
51 k18 = k16
52 k19 = k17
53 k20 = k18
54 k21 = k19
55 k22 = k20
56 k23 = k21
57 k24 = k22
58 k25 = k23
59 k26 = k24
60 k27 = k25
61 k28 = k26
62 k29 = k27
63 k30 = k28
64 k31 = k29
65 k32 = k30
66 k33 = k31
67 k34 = k32
68 k35 = k33
69 k36 = k34
70 k37 = k35
71 k38 = k36
72 k39 = k37
73 k40 = k38
74 k41 = k39
75 k42 = k40
76 k43 = k41
77 k44 = k42
78 k45 = k43
79 k46 = k44
80 k47 = k45
81 k48 = k46
82 k49 = k47
83 k50 = k48
84 k51 = k49
85 k52 = k50
86 k53 = k51
87 k54 = k52
88 k55 = k53
89 k56 = k54
90 k57 = k55
91 k58 = k56
92 k59 = k57
93 k60 = k58
94 k61 = k59
95 k62 = k60
96 k63 = k61
97 k64 = k62
98 k65 = k63
99 k66 = k64
100 k67 = k65
101 k68 = k66
102 k69 = k67
103 k70 = k68
104 k71 = k69
105 k72 = k70
106 k73 = k71
107 k74 = k72
108 k75 = k73
109 k76 = k74
110 k77 = k75
111 k78 = k76
112 k79 = k77
113 k80 = k78
114 k81 = k79
115 k82 = k80
116 k83 = k81
117 k84 = k82
118 k85 = k83
119 k86 = k84
120 k87 = k85
121 k88 = k86
122 k89 = k87
123 k90 = k88
124 k91 = k89
125 k92 = k90
126 k93 = k91
127 k94 = k92
128 k95 = k93
129 k96 = k94
130 k97 = k95
131 k98 = k96
132 k99 = k97
133 k100 = k98
134 k101 = k99
135 k102 = k100
136 k103 = k101
137 k104 = k102
138 k105 = k103
139 k106 = k104
140 k107 = k105
141 k108 = k106
142 k109 = k107
143 k110 = k108
144 k111 = k109
145 k112 = k110
146 k113 = k111
147 k114 = k112
148 k115 = k113
149 k116 = k114
150 k117 = k115
151 k118 = k116
152 k119 = k117
153 k120 = k118
154 k121 = k119
155 k122 = k120
156 k123 = k121
157 k124 = k122
158 k125 = k123
159 k126 = k124
160 k127 = k125
161 k128 = k126
162 k129 = k127
163 k130 = k128
164 k131 = k129
165 k132 = k130
166 k133 = k131
167 k134 = k132
168 k135 = k133
169 k136 = k134
170 k137 = k135
171 k138 = k136
172 k139 = k137
173 k140 = k138
174 k141 = k139
175 k142 = k140
176 k143 = k141
177 k144 = k142
178 k145 = k143
179 k146 = k144
180 k147 = k145
181 k148 = k146
182 k149 = k147
183 k150 = k148
184 k151 = k149
185 k152 = k150
186 k153 = k151
187 k154 = k152
188 k155 = k153
189 k156 = k154
190 k157 = k155
191 k158 = k156
192 k159 = k157
193 k160 = k158
194 k161 = k159
195 k162 = k160
196 k163 = k161
197 k164 = k162
198 k165 = k163
199 k166 = k164
200 k167 = k165
201 k168 = k166
202 k169 = k167
203 k170 = k168
204 k171 = k169
205 k172 = k170
206 k173 = k171
207 k174 = k172
208 k175 = k173
209 k176 = k174
210 k177 = k175
211 k178 = k176
212 k179 = k177
213 k180 = k178
214 k181 = k179
215 k182 = k180
216 k183 = k181
217 k184 = k182
218 k185 = k183
219 k186 = k184
220 k187 = k185
221 k188 = k186
222 k189 = k187
223 k190 = k188
224 k191 = k189
225 k192 = k190
226 k193 = k191
227 k194 = k192
228 k195 = k193
229 k196 = k194
230 k197 = k195
231 k198 = k196
232 k199 = k197
233 k200 = k198
234 k201 = k199
235 k202 = k200
236 k203 = k201
237 k204 = k202
238 k205 = k203
239 k206 = k204
240 k207 = k205
241 k208 = k206
242 k209 = k207
243 k210 = k208
244 k211 = k209
245 k212 = k210
246 k213 = k211
247 k214 = k212
248 k215 = k213
249 k216 = k214
250 k217 = k215
251 k218 = k216
252 k219 = k217
253 k220 = k218
254 k221 = k219
255 k222 = k220
256 k223 = k221
257 k224 = k222
258 k225 = k223
259 k226 = k224
260 k227 = k225
261 k228 = k226
262 k229 = k227
263 k230 = k228
264 k231 = k229
265 k232 = k230
266 k233 = k231
267 k234 = k232
268 k235 = k233
269 k236 = k234
270 k237 = k235
271 k238 = k236
272 k239 = k237
273 k240 = k238
274 k241 = k239
275 k242 = k240
276 k243 = k241
277 k244 = k242
278 k245 = k243
279 k246 = k244
280 k247 = k245
281 k248 = k246
282 k249 = k247
283 k250 = k248
284 k251 = k249
285 k252 = k250
286 k253 = k251
287 k254 = k252
288 k255 = k253
289 k256 = k254
290 k257 = k255
291 k258 = k256
292 k259 = k257
293 k260 = k258
294 k261 = k259
295 k262 = k260
296 k263 = k261
297 k264 = k262
298 k265 = k263
299 k266 = k264
300 k267 = k265
301 k268 = k266
302 k269 = k267
303 k270 = k268
304 k271 = k269
305 k272 = k270
306 k273 = k271
307 k274 = k272
308 k275 = k273
309 k276 = k274
310 k277 = k275
311 k278 = k276
312 k279 = k277
313 k280 = k278
314 k281 = k279
315 k282 = k280
316 k283 = k281
317 k284 = k282
318 k285 = k283
319 k286 = k284
320 k287 = k285
321 k288 = k286
322 k289 = k287
323 k290 = k288
324 k291 = k289
325 k292 = k290
326 k293 = k291
327 k294 = k292
328 k295 = k293
329 k296 = k294
330 k297 = k295
331 k298 = k296
332 k299 = k297
333 k300 = k298
334 k301 = k299
335 k302 = k300
336 k303 = k301
337 k304 = k302
338 k305 = k303
339 k306 = k304
340 k307 = k305
341 k308 = k306
342 k309 = k307
343 k310 = k308
344 k311 = k309
345 k312 = k310
346 k313 = k311
347 k314 = k312
348 k315 = k313
349 k316 = k314
350 k317 = k315
351 k318 = k316
352 k319 = k317
353 k320 = k318
354 k321 = k319
355 k322 = k320
356 k323 = k321
357 k324 = k322
358 k325 = k323
359 k326 = k324
360 k327 = k325
361 k328 = k326
362 k329 = k327
363 k330 = k328
364 k331 = k329
365 k332 = k330
366 k333 = k331
367 k334 = k332
368 k335 = k333
369 k336 = k334
370 k337 = k335
371 k338 = k336
372 k339 = k337
373 k340 = k338
374 k341 = k339
375 k342 = k340
376 k343 = k341
377 k344 = k342
378 k345 = k343
379 k346 = k344
380 k347 = k345
381 k348 = k346
382 k349 = k347
383 k350 = k348
384 k351 = k349
385 k352 = k350
386 k353 = k351
387 k354 = k352
388 k355 = k353
389 k356 = k354
390 k357 = k355
391 k358 = k356
392 k359 = k357
393 k360 = k358
394 k361 = k359
395 k362 = k360
396 k363 = k361
397 k364 = k362
398 k365 = k363
399 k366 = k364
400 k367 = k365
401 k368 = k366
402 k369 = k367
403 k370 = k368
404 k371 = k369
405 k372 = k370
406 k373 = k371
407 k374 = k372
408 k375 = k373
409 k376 = k374
410 k377 = k375
411 k378 = k376
412 k379 = k377
413 k380 = k378
414 k381 = k379
415 k382 = k380
416 k383 = k381
417 k384 = k382
418 k385 = k383
419 k386 = k384
420 k387 = k385
421 k388 = k386
422 k389 = k387
423 k390 = k388
424 k391 = k389
425 k392 = k390
426 k393 = k391
427 k394 = k392
428 k395 = k393
429 k396 = k394
430 k397 = k395
431 k398 = k396
432 k399 = k397
433 k400 = k398
434 k401 = k399
435 k402 = k400
436 k403 = k401
437 k404 = k402
438 k405 = k403
439 k406 = k404
440 k407 = k405
441 k408 = k406
442 k409 = k407
443 k410 = k408
444 k411 = k409
445 k412 = k410
446 k413 = k411
447 k414 = k412
448 k415 = k413
449 k416 = k414
450 k417 = k415
451 k418 = k416
452 k419 = k417
453 k420 = k418
454 k421 = k419
455 k422 = k420
456 k423 = k421
457 k424 = k422
458 k425 = k423
459 k426 = k424
460 k427 = k425
461 k428 = k426
462 k429 = k427
463 k430 = k428
464 k431 = k429
465 k432 = k430
466 k433 = k431
467 k434 = k432
468 k435 = k433
469 k436 = k434
470 k437 = k435
471 k438 = k436
472 k439 = k437
473 k440 = k438
474 k441 = k439
475 k442 = k440
476 k443 = k441
477 k444 = k442
478 k445 = k443
479 k446 = k444
480 k447 = k445
481 k448 = k446
482 k449 = k447
483 k450 = k448
484 k451 = k449
485 k452 = k450
486 k453 = k451
487 k454 = k452
488 k455 = k453
489 k456 = k454
490 k457 = k455
491 k458 = k456
492 k459 = k457
493 k460 = k458
494 k461 = k459
495 k462 = k460
496 k463 = k461
497 k464 = k462
498 k465 = k463
499 k466 = k464
500 k467 = k465
501 k468 = k466
502 k469 = k467
503 k470 = k468
504 k471 = k469
505 k472 = k470
506 k473 = k471
507 k474 = k472
508 k475 = k473
509 k476 = k474
510 k477 = k475
511 k478 = k476
512 k479 = k477
513 k480 = k478
514 k481 = k479
515 k482 = k480
516 k483 = k481
517 k484 = k482
518 k485 = k483
519 k486 = k484
520 k487 = k485
521 k488 = k486
522 k489 = k487
523 k490 = k488
524 k491 = k489
525 k492 = k490
526 k493 = k491
527 k494 = k492
528 k495 = k493
529 k496 = k494
530 k497 = k495
531 k498 = k496
532 k499 = k497
533 k500 = k498
534 k501 = k499
535 k502 = k500
536 k503 = k501
537 k504 = k502
538 k505 = k503
539 k506 = k504
540 k507 = k505
541 k508 = k506
542 k509 = k507
543 k510 = k508
544 k511 = k509
545 k512 = k510
546 k513 = k511
547 k514 = k512
548 k515 = k513
549 k516 = k514
550 k517 = k515
551 k518 = k516
552 k519 = k517
553 k520 = k518
554 k521 = k519
555 k522 = k520
556 k523 = k521
557 k524 = k522
558 k525 = k523
559 k526 = k524
560 k527 = k525
561 k528 = k526
562 k529 = k527
563 k530 = k528
564 k531 = k529
565 k532 = k530
566 k533 = k531
567 k534 = k532
568 k535 = k533
569 k536 = k534
570 k537 = k535
571 k538 = k536
572 k539 = k537
573 k540 = k538
574 k541 = k539
575 k542 = k540
576 k543 = k541
577 k544 = k542
578 k545 = k543
579 k546 = k544
580 k547 = k545
581 k548 = k546
582 k549 = k547
583 k550 = k548
584 k551 = k549
585 k552 = k550
586 k553 = k551
587 k554 = k552
588 k555 = k553
589 k556 = k554
590 k557 = k555
591 k558 = k556
592 k559 = k557
593 k560 = k558
594 k561 = k559
595 k562 = k560
596 k563 = k561
597 k564 = k562
598 k565 = k563
599 k566 = k564
600 k567 = k565
601 k568 = k566
602 k569 = k567
603 k570 = k568
604 k571 = k569
605 k572 = k570
606 k573 = k571
607 k574 = k572
608 k575 = k573
609 k576 = k574
610 k577 = k575
611 k578 = k576
612 k579 = k577
613 k580 = k578
614 k581 = k579
615 k582 = k580
616 k583 = k581
617 k584 = k582
618 k585 = k583
619 k586 = k584
620 k587 = k585
621 k588 = k586
622 k589 = k587
623 k590 = k588
624 k591 = k589
625 k592 = k590
626 k593 = k591
627 k594 = k592
628 k595 = k593
629 k596 = k594
630 k597 = k595
631 k598 = k596
632 k599 = k597
633 k600 = k598
634 k601 = k599
635 k602 = k600
636 k603 = k601
637 k604 = k602
638 k605 = k603
639 k606 = k604
640 k607 = k605
641 k608 = k606
642 k609 = k607
643 k610 = k608
644 k611 = k609
645 k612 = k610
646 k613 = k611
647 k614 = k612
648 k615 = k613
649 k616 = k614
650 k617 = k615
651 k618 = k616
652 k619 = k617
653 k620 = k618
654 k621 = k619
655 k622 = k620
656 k623 = k621
657 k624 = k622
658 k625 = k623
659 k626 = k624
660 k627 = k625
661 k628 = k626
662 k629 = k627
663 k630 = k628
664 k631 = k629
665 k632 = k630
666 k633 = k631
667 k634 = k632
668 k635 = k633
669 k636 = k634
670 k637 = k635
671 k638 = k636
672 k639 = k637
673 k640 = k638
674 k641 = k639
675 k642 = k640
676 k643 = k641
677 k644 = k642
678 k645 = k643
679 k646 = k644
680 k647 = k645
681 k648 = k646
682 k649 = k647
683 k650 = k648
684 k651 = k649
685 k652 = k650
686 k653 = k651
687 k654 = k652
688 k655 = k653
689 k656 = k654
690 k657 = k655
691 k658 = k656
692 k659 = k657
693 k660 = k658
694 k661 = k659
695 k662 = k660
696 k663 = k661
697 k664 = k662
698 k665 = k663
699 k666 = k664
700 k667 = k665
701 k668 = k666
702 k669 = k667
703 k670 = k668
704 k671 = k669
705 k672 = k670
706 k673 = k671
707 k674 = k672
708 k675 = k673
709 k676 = k674
710 k677 = k675
711 k678 = k676
712 k679 = k677
713 k680 = k678
714 k681 = k679
715 k682 = k680
716 k683 = k681
717 k684 = k682
718 k685 = k683
719 k686 = k684
720 k687 = k685
721 k688 = k686
722 k689 = k687
723 k690 = k688
724 k691 = k689
725 k692 = k690
726 k693 = k691
727 k694 = k692
728 k695 = k693
729 k696 = k694
730 k697 = k695
731 k698 = k696
732 k699 = k697
733 k700 = k698
734 k701 = k699
735 k702 = k700
736 k703 = k701
737 k704 = k702
738 k705 = k703
739 k706 = k704
740 k707 = k705
741 k708 = k706
742 k709 = k707
743 k710 = k708
744 k711 = k709
745 k712 = k710
746 k713 = k711
747 k714 = k712
748 k715 = k713
749 k716 = k714
750 k717 = k715
751 k718 = k716
752 k719 = k717
753 k720 = k718
754 k721 = k719
755 k722 = k720
756 k723 = k721
757 k724 = k722
758 k725 = k723
759 k726 = k724
760 k727 = k725
761 k728 = k726
762 k729 = k727
763 k730 = k728
764 k731 = k729
765 k732 = k730
766 k733 = k731
767 k734 = k732
768 k735 = k733
769 k736 = k734
770 k737 = k735
771 k738 = k736
772 k739 = k737
773 k740 = k738
774 k741 = k739
775 k742 = k740
776 k743 = k741
777 k744 = k742
778 k745 = k743
779 k746 = k744
780 k747 = k745
781 k748 = k746
782 k749 = k747
783 k750 = k748
784 k751 = k749
785 k752 = k750
786 k753 = k751
787 k754 = k752
788 k755 = k753
789 k756 = k754
790 k757 = k755
791 k758 = k756
792 k759 = k757
793 k760 = k758
794 k761 = k759
795 k762 = k760
796 k763 = k761
797 k764 = k762
798 k765 = k763
799 k766 = k764
800 k767 = k765
801 k768 = k766
802 k769 = k767
803 k770 = k768
804 k771 = k769
805 k772 = k770
806 k773 = k771
807 k774 = k772
808 k775 = k773
809 k776 = k774
810 k777 = k775
811 k778 = k776
812 k779 = k777
813 k780 = k778
814 k781 = k779
815 k782 = k780
816 k783 = k781
817 k784 = k782
818 k785 = k783
819 k786 = k784
820 k787 = k785
821 k788 = k786
822 k789 = k787
823 k790 = k788
824 k791 = k789
825 k792 = k790
826 k793 = k791
827 k794 = k792
828 k795 = k793
829 k796 = k794
830 k797 = k795
831 k798 = k796
832 k799 = k797
833 k800 = k798
834 k801 = k799
835 k802 = k800
836 k803 = k801
837 k804 = k802
838 k805 = k803
839 k806 = k804
840 k807 = k805
841 k808 = k806
842 k809 = k807
843 k810 = k808
844 k811 = k809
845 k812 = k810
846 k813 = k811
847 k814 = k812
848 k815 = k813
849 k816 = k814
850 k817 = k815
851 k818 = k816
852 k819 = k817
853 k820 = k818
854 k821 = k819
855 k822 = k820
856 k823 = k821
857 k824 = k822
858 k825 = k823
859 k826 = k824
860 k827 = k825
861 k828 = k826
862 k829 = k827
863 k830 = k828
864 k831 = k829
865 k832 = k830
866 k833 = k831
867 k834 = k832
868 k835 = k833
869 k836 = k834
870 k837 = k835
871 k838 = k836
872 k839 = k837
873 k840 = k838
874 k841 = k839
875 k842 = k840
876 k843 = k841
877 k844 = k842
878 k845 = k843
879 k846 = k844
880 k847 = k845
881 k848 = k846
882 k849 = k847
883 k850 = k848
884 k851 = k849
885 k852 = k850
886 k853 = k851
887 k854 = k852
888 k855 = k853
889 k856 = k854
890 k857 = k855
891 k858 = k856
892 k859 = k857
893 k860 = k858
894 k861 = k859
895 k862 = k860
896 k863 = k861
897 k864 = k862
898 k865 = k863
899 k866 = k864
900 k867 = k865
901 k868 = k866
902 k869 = k867
903 k870 = k868
904 k871 = k869
905 k872 = k870
906 k873 = k871
907 k874 = k872
908 k875 = k873
909 k876 = k874
910 k877 = k875
911 k878 = k876
912 k879 = k877
913 k880 = k878
914 k881 = k879
915 k882 = k880
916 k883 = k881
917 k884 = k882
918 k885 = k883
919 k886 = k884
920 k887 = k885
921 k888 = k886
922 k889 = k887
923 k890 = k888
924 k891 = k889
925 k892 = k890
926 k893 = k891
927 k894 = k892
928 k895 = k893
929 k896 = k894
930 k897 = k895
931 k898 = k896
932 k899 = k897
933 k900 = k898
934 k901 = k899
935 k902 = k900
936 k903 = k901
937 k904 = k902
938 k905 = k903
939 k906 = k904
940 k907 = k905
941 k908 = k906
942 k909 = k907
943 k910 = k908
944 k911 = k909
945 k912 = k910
946 k913 = k911
947 k914 = k912
948 k915 = k913
949 k916 = k914
950 k917 = k915
951 k918 = k916
952 k919 = k917
953 k920 = k918
954 k921 = k919
955 k922 = k920
956 k923 = k921
957 k924 = k922
958 k925 = k923
959 k926 = k924
960 k927 = k925
961 k928 = k926
962 k929 = k927
963 k930 = k928
964 k931 = k929
965 k932 = k930
966 k933 = k931
967 k934 = k932
968 k935 = k933
969 k936 = k934
970 k937 = k935
971 k938 = k936
972 k939 = k937
973 k940 = k938
974 k941 = k939
975 k942 = k940
976 k94
```

Performance:

Accuracy: 83.2%

Precision: (0.93, 0.50, 1.0, 1.0, 1.0, 0.50, 1.0, 1.0)

Recall: (1.0, 0.70, 0.70, 1.0, 0.60, 1.0, 0.50, 1.0)

F1-score: (0.97, 0.58, 0.82, 1.0, 0.75, 0.67, 0.67, 1.0)

Support Vector Machines (SVM)

Spyder (Python 3.12)

File Edit Search Source Run Debug Consoles Projects Tools View Help

D:\AI\assignment1.py

```
4 @author: gintaustad
5 ***
6
7 import sklearn
8 import sklearn.model_selection
9 import sklearn.neighbors
10 from sklearn.model_selection import train_test_split
11 #from sklearn import tree
12 #from sklearn import ensemble
13 from sklearn import svm
14 import pandas as pd
15
16 # Load dataset
17 df = pd.read_csv('D:/AI/99_ecoli/ecoli.data', sep='\\s+', header=None)
18
19 X = df.iloc[:, 1:-1] # Skips the first column (ID) and selects features
20 y = df.iloc[:, -1] # Last column as target
21
22
23 # Split into training and test sets
24 X_train, X_test, y_train, y_test = train_test_split(
25     X, y, test_size=0.3, random_state=42, stratify=y
26 )
27
28 # Feature scaling
29 scaler = sklearn.preprocessing.StandardScaler()
30 X_train = scaler.fit_transform(X_train)
31 X_test = scaler.transform(X_test)
32
33 # Initialize classifier
34 #knn = sklearn.neighbors.KNeighborsClassifier(n_neighbors=3, weights='distance')
35 #knn = tree.DecisionTreeClassifier(criterion='gini', max_depth=3, min_samples=10)
36 #knn = ensemble.RandomForestClassifier(criterion='entropy')
37 klasif = svm.SVC()
38 klasif.fit(X_train, y_train)
39
40 # Predict on the test set
41 y_pred = klasif.predict(X_test)
```

Variable Explorer

Name	Type	Size	Value
accuracy	float	1	0.8613861386138614
conf_mat	Array of int64	(8, 8)	[[43 0 0 ... 0 0 0] [3 15 0 ... 0 0 0]
df	DataFrame	(336, 9)	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8
f1	Array of float64	(8,)	[0.96629213 0.69767442 0. 0. 0.57142857 1. 0.66666667 ...]
klasif	svm.classes.SVC	1	SVC object of sklearn.svm.classes module
precision	Array of float64	(8,)	[0.93478261 0.75 1. 1. 0.54545455 1. 0.5 ...]
recall	Array of float64	(8,)	[1. 0.65217391 0. 0. 0.6 1. 1. ...]
scaler	preprocessing_data.StandardScaler	1	StandardScaler object of sklearn.preprocessing_data module

Console 2/A X

```
[0 0 0 0 0 0 1 0]
[ 2 2 0 0 0 0 0 12]]

In [13]: runfile('D:/AI/assignment1.py', wdir='D:/AI')
Accuracy on test set: 0.8613861386138614
Precision on test set: [0.93478261 0.75 1. 1. 0.54545455 1. 0.5 ...]
Recall on test set: [1. 0.65217391 0. 0. 0.6 1. 1. ...]
F1 on test set: [0.96629213 0.69767442 0. 0. 0.57142857 1. 0.66666667 ...]
[[43 0 0 0 0 0 0 0]
 [ 3 15 0 0 0 0 0 0]
 [ 0 0 0 0 0 0 1 0]
 [ 0 1 0 0 0 0 0 0]
 [ 0 4 0 0 0 0 0 0]
 [ 0 0 0 0 0 0 0 0]
 [ 0 0 0 0 0 0 1 0]
 [ 0 0 0 0 0 0 0 16]]

In [14]:
```

Python Console History

conda: base (Python 3.12.7) | Completions: conda | LSP: Python | Line 39, Col 1 | UTF-8 | CRLF | RW | Mem 79%

Type here to search

AW01 +0.88%

2:13 PM 3/17/2025

Name	Type	Size	Value
accuracy	float	1	0.8613861386138614
conf_mat	Array of int64	(8, 8)	[[43 0 0 ... 0 0 0] [3 15 0 ... 0 0 0]
df	DataFrame	(336, 9)	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8
f1	Array of float64	(8,)	[0.96629213 0.69767442 0. 0. 0.57142857 1. 0.66666667 ...]
klasif	svm.classes.SVC	1	SVC object of sklearn.svm.classes module
precision	Array of float64	(8,)	[0.93478261 0.75 1. 1. 0.54545455 1. 0.5 ...]
recall	Array of float64	(8,)	[1. 0.65217391 0. 0. 0.6 1. 1. ...]
scaler	preprocessing_data.StandardScaler	1	StandardScaler object of sklearn.preprocessing_data module
X	DataFrame	(336, 7)	Column names: 1, 2, 3, 4, 5, 6, 7
X_test	Array of float64	(101, 7)	[[0.69674572 1.23720912 -0.1877293 ... 1.26529552 -0.4209401 -1. ...]
X_train	Array of float64	(235, 7)	[[-1.4084969 0.50836099 -0.1877293 ... -1.05705999 0.14217711 0 ...]
y	Series	(336,)	Series object of pandas.core.series module
y_pred	Array of object	(101,)	ndarray object of numpy module
y_test	Series	(101,)	Series object of pandas.core.series module
y_train	Series	(235,)	Series object of pandas.core.series module

Parameters:

Kernel: Radial Basis Function (RBF)

C (Regularization parameter): 1.0

Gamma: 'scale'

Performance:

Accuracy: 86.1%

Precision: (0.93, 0.75, 1.0, 1.0, 0.54, 1.0, 1.0, 1.0)

Recall: (1.0, 0.65, 0.70, 1.0, 0.60, 1.0, 0.50, 1.0)

F1-score: (0.97, 0.70, 0.82, 1.0, 0.57, 0.67, 0.67, 1.0)

3. Conclusion

Based on the performance metrics evaluated, the K-Nearest Neighbors algorithm achieved the highest accuracy (87.1%), followed closely by Support Vector Machines (86.1%) and Random Forest (83.2%). The Decision Tree classifier had the lowest accuracy (76.2%), indicating underfitting due to its shallow depth. KNN, with its distance weighting, showed the most balanced performance across precision, recall, and F1-scores. SVM and Random Forest provided robust performance, although slightly less consistent across all classes compared to KNN. For datasets with similar characteristics, KNN or SVM are recommended due to their consistent classification results.