

Exercise-9

November 2, 2025

```
[1]: import pandas as pd
df=pd.read_csv('Social_Network_Ads.csv')
```

```
[3]: import numpy as np
import pandas as pd
```

```
[4]: features=df.iloc[:,[2,3]].values
label=df.iloc[:,4].values
features
```

```
[4]: array([[ 19, 19000],
 [ 35, 20000],
 [ 26, 43000],
 [ 27, 57000],
 [ 19, 76000],
 [ 27, 58000],
 [ 27, 84000],
 [ 32, 150000],
 [ 25, 33000],
 [ 35, 65000],
 [ 26, 80000],
 [ 26, 52000],
 [ 20, 86000],
 [ 32, 18000],
 [ 18, 82000],
 [ 29, 80000],
 [ 47, 25000],
 [ 45, 26000],
 [ 46, 28000],
 [ 48, 29000],
 [ 45, 22000],
 [ 47, 49000],
 [ 48, 41000],
 [ 45, 22000],
 [ 46, 23000],
 [ 47, 20000],
 [ 49, 28000],
 [ 47, 30000],
```

[29, 43000],
[31, 18000],
[31, 74000],
[27, 137000],
[21, 16000],
[28, 44000],
[27, 90000],
[35, 27000],
[33, 28000],
[30, 49000],
[26, 72000],
[27, 31000],
[27, 17000],
[33, 51000],
[35, 108000],
[30, 15000],
[28, 84000],
[23, 20000],
[25, 79000],
[27, 54000],
[30, 135000],
[31, 89000],
[24, 32000],
[18, 44000],
[29, 83000],
[35, 23000],
[27, 58000],
[24, 55000],
[23, 48000],
[28, 79000],
[22, 18000],
[32, 117000],
[27, 20000],
[25, 87000],
[23, 66000],
[32, 120000],
[59, 83000],
[24, 58000],
[24, 19000],
[23, 82000],
[22, 63000],
[31, 68000],
[25, 80000],
[24, 27000],
[20, 23000],
[33, 113000],
[32, 18000],

[34, 112000],
[18, 52000],
[22, 27000],
[28, 87000],
[26, 17000],
[30, 80000],
[39, 42000],
[20, 49000],
[35, 88000],
[30, 62000],
[31, 118000],
[24, 55000],
[28, 85000],
[26, 81000],
[35, 50000],
[22, 81000],
[30, 116000],
[26, 15000],
[29, 28000],
[29, 83000],
[35, 44000],
[35, 25000],
[28, 123000],
[35, 73000],
[28, 37000],
[27, 88000],
[28, 59000],
[32, 86000],
[33, 149000],
[19, 21000],
[21, 72000],
[26, 35000],
[27, 89000],
[26, 86000],
[38, 80000],
[39, 71000],
[37, 71000],
[38, 61000],
[37, 55000],
[42, 80000],
[40, 57000],
[35, 75000],
[36, 52000],
[40, 59000],
[41, 59000],
[36, 75000],
[37, 72000],

[40, 75000],
[35, 53000],
[41, 51000],
[39, 61000],
[42, 65000],
[26, 32000],
[30, 17000],
[26, 84000],
[31, 58000],
[33, 31000],
[30, 87000],
[21, 68000],
[28, 55000],
[23, 63000],
[20, 82000],
[30, 107000],
[28, 59000],
[19, 25000],
[19, 85000],
[18, 68000],
[35, 59000],
[30, 89000],
[34, 25000],
[24, 89000],
[27, 96000],
[41, 30000],
[29, 61000],
[20, 74000],
[26, 15000],
[41, 45000],
[31, 76000],
[36, 50000],
[40, 47000],
[31, 15000],
[46, 59000],
[29, 75000],
[26, 30000],
[32, 135000],
[32, 100000],
[25, 90000],
[37, 33000],
[35, 38000],
[33, 69000],
[18, 86000],
[22, 55000],
[35, 71000],
[29, 148000],

```

[ 29, 47000],
[ 21, 88000],
[ 34, 115000],
[ 26, 118000],
[ 34, 43000],
[ 34, 72000],
[ 23, 28000],
[ 35, 47000],
[ 25, 22000],
[ 24, 23000],
[ 31, 34000],
[ 26, 16000],
[ 31, 71000],
[ 32, 117000],
[ 33, 43000],
[ 33, 60000],
[ 31, 66000],
[ 20, 82000],
[ 33, 41000],
[ 35, 72000],
[ 28, 32000],
[ 24, 84000],
[ 19, 26000],
[ 29, 43000],
[ 19, 70000],
[ 28, 89000],
[ 34, 43000],
[ 30, 79000],
[ 20, 36000],
[ 26, 80000],
[ 35, 22000],
[ 35, 39000],
[ 49, 74000],
[ 39, 134000],
[ 41, 71000],
[ 58, 101000],
[ 47, 47000],
[ 55, 130000],
[ 52, 114000],
[ 40, 142000],
[ 46, 22000],
[ 48, 96000],
[ 52, 150000],
[ 59, 42000],
[ 35, 58000],
[ 47, 43000],
[ 60, 108000],

```

```

[ 49, 65000],
[ 40, 78000],
[ 46, 96000],
[ 59, 143000],
[ 41, 80000],
[ 35, 91000],
[ 37, 144000],
[ 60, 102000],
[ 35, 60000],
[ 37, 53000],
[ 36, 126000],
[ 56, 133000],
[ 40, 72000],
[ 42, 80000],
[ 35, 147000],
[ 39, 42000],
[ 40, 107000],
[ 49, 86000],
[ 38, 112000],
[ 46, 79000],
[ 40, 57000],
[ 37, 80000],
[ 46, 82000],
[ 53, 143000],
[ 42, 149000],
[ 38, 59000],
[ 50, 88000],
[ 56, 104000],
[ 41, 72000],
[ 51, 146000],
[ 35, 50000],
[ 57, 122000],
[ 41, 52000],
[ 35, 97000],
[ 44, 39000],
[ 37, 52000],
[ 48, 134000],
[ 37, 146000],
[ 50, 44000],
[ 52, 90000],
[ 41, 72000],
[ 40, 57000],
[ 58, 95000],
[ 45, 131000],
[ 35, 77000],
[ 36, 144000],
[ 55, 125000],

```

[35, 72000],
[48, 90000],
[42, 108000],
[40, 75000],
[37, 74000],
[47, 144000],
[40, 61000],
[43, 133000],
[59, 76000],
[60, 42000],
[39, 106000],
[57, 26000],
[57, 74000],
[38, 71000],
[49, 88000],
[52, 38000],
[50, 36000],
[59, 88000],
[35, 61000],
[37, 70000],
[52, 21000],
[48, 141000],
[37, 93000],
[37, 62000],
[48, 138000],
[41, 79000],
[37, 78000],
[39, 134000],
[49, 89000],
[55, 39000],
[37, 77000],
[35, 57000],
[36, 63000],
[42, 73000],
[43, 112000],
[45, 79000],
[46, 117000],
[58, 38000],
[48, 74000],
[37, 137000],
[37, 79000],
[40, 60000],
[42, 54000],
[51, 134000],
[47, 113000],
[36, 125000],
[38, 50000],

[42, 70000],
[39, 96000],
[38, 50000],
[49, 141000],
[39, 79000],
[39, 75000],
[54, 104000],
[35, 55000],
[45, 32000],
[36, 60000],
[52, 138000],
[53, 82000],
[41, 52000],
[48, 30000],
[48, 131000],
[41, 60000],
[41, 72000],
[42, 75000],
[36, 118000],
[47, 107000],
[38, 51000],
[48, 119000],
[42, 65000],
[40, 65000],
[57, 60000],
[36, 54000],
[58, 144000],
[35, 79000],
[38, 55000],
[39, 122000],
[53, 104000],
[35, 75000],
[38, 65000],
[47, 51000],
[47, 105000],
[41, 63000],
[53, 72000],
[54, 108000],
[39, 77000],
[38, 61000],
[38, 113000],
[37, 75000],
[42, 90000],
[37, 57000],
[36, 99000],
[60, 34000],
[54, 70000],


```
[ 41, 72000],
[ 40, 71000],
[ 42, 54000],
[ 43, 129000],
[ 53, 34000],
[ 47, 50000],
[ 42, 79000],
[ 42, 104000],
[ 59, 29000],
[ 58, 47000],
[ 46, 88000],
[ 38, 71000],
[ 54, 26000],
[ 60, 46000],
[ 60, 83000],
[ 39, 73000],
[ 59, 130000],
[ 37, 80000],
[ 46, 32000],
[ 46, 74000],
[ 42, 53000],
[ 41, 87000],
[ 58, 23000],
[ 42, 64000],
[ 48, 33000],
[ 44, 139000],
[ 49, 28000],
[ 57, 33000],
[ 56, 60000],
[ 49, 39000],
[ 39, 71000],
[ 47, 34000],
[ 48, 35000],
[ 48, 33000],
[ 47, 23000],
[ 45, 45000],
[ 60, 42000],
[ 39, 59000],
[ 46, 41000],
[ 51, 23000],
[ 50, 20000],
[ 36, 33000],
[ 49, 36000]])
```

```
[5]: from sklearn.model_selection import train_test_split
      from sklearn.linear_model import LogisticRegression
```

```
[6]: for i in range(1, 401):
    x_train, x_test, y_train, y_test = train_test_split(features, label,
    ↪test_size=0.2, random_state=i)

    model = LogisticRegression()
    model.fit(x_train, y_train)

    train_score = model.score(x_train, y_train)
    test_score = model.score(x_test, y_test)

    if test_score > train_score:
        print("Test {:.3f} Train {:.3f} Random State {}".format(test_score,
    ↪train_score, i))
```

```
Test 0.900 Train 0.841 Random State 4
Test 0.863 Train 0.850 Random State 5
Test 0.863 Train 0.859 Random State 6
Test 0.887 Train 0.838 Random State 7
Test 0.863 Train 0.838 Random State 9
Test 0.900 Train 0.841 Random State 10
Test 0.863 Train 0.856 Random State 14
Test 0.850 Train 0.844 Random State 15
Test 0.863 Train 0.856 Random State 16
Test 0.875 Train 0.834 Random State 18
Test 0.850 Train 0.844 Random State 19
Test 0.875 Train 0.844 Random State 20
Test 0.863 Train 0.834 Random State 21
Test 0.875 Train 0.841 Random State 22
Test 0.875 Train 0.841 Random State 24
Test 0.850 Train 0.834 Random State 26
Test 0.850 Train 0.841 Random State 27
Test 0.863 Train 0.834 Random State 30
Test 0.863 Train 0.856 Random State 31
Test 0.875 Train 0.853 Random State 32
Test 0.863 Train 0.844 Random State 33
Test 0.875 Train 0.831 Random State 35
Test 0.863 Train 0.853 Random State 36
Test 0.887 Train 0.841 Random State 38
Test 0.875 Train 0.838 Random State 39
Test 0.887 Train 0.838 Random State 42
Test 0.875 Train 0.847 Random State 46
Test 0.912 Train 0.831 Random State 47
Test 0.875 Train 0.831 Random State 51
Test 0.900 Train 0.844 Random State 54
Test 0.850 Train 0.844 Random State 57
Test 0.875 Train 0.844 Random State 58
Test 0.925 Train 0.838 Random State 61
Test 0.887 Train 0.834 Random State 65
```

Test 0.887 Train 0.841 Random State 68
Test 0.900 Train 0.831 Random State 72
Test 0.887 Train 0.838 Random State 75
Test 0.925 Train 0.825 Random State 76
Test 0.863 Train 0.841 Random State 77
Test 0.863 Train 0.859 Random State 81
Test 0.875 Train 0.838 Random State 82
Test 0.887 Train 0.838 Random State 83
Test 0.863 Train 0.853 Random State 84
Test 0.863 Train 0.841 Random State 85
Test 0.863 Train 0.841 Random State 87
Test 0.875 Train 0.847 Random State 88
Test 0.912 Train 0.838 Random State 90
Test 0.863 Train 0.850 Random State 95
Test 0.875 Train 0.850 Random State 99
Test 0.850 Train 0.841 Random State 101
Test 0.850 Train 0.841 Random State 102
Test 0.900 Train 0.825 Random State 106
Test 0.863 Train 0.841 Random State 107
Test 0.850 Train 0.834 Random State 109
Test 0.850 Train 0.841 Random State 111
Test 0.912 Train 0.841 Random State 112
Test 0.863 Train 0.850 Random State 115
Test 0.863 Train 0.841 Random State 116
Test 0.875 Train 0.834 Random State 119
Test 0.912 Train 0.828 Random State 120
Test 0.863 Train 0.859 Random State 125
Test 0.850 Train 0.847 Random State 128
Test 0.875 Train 0.850 Random State 130
Test 0.900 Train 0.844 Random State 133
Test 0.925 Train 0.834 Random State 134
Test 0.863 Train 0.850 Random State 135
Test 0.875 Train 0.831 Random State 138
Test 0.863 Train 0.850 Random State 141
Test 0.850 Train 0.847 Random State 143
Test 0.850 Train 0.847 Random State 146
Test 0.850 Train 0.844 Random State 147
Test 0.863 Train 0.850 Random State 148
Test 0.875 Train 0.838 Random State 150
Test 0.887 Train 0.831 Random State 151
Test 0.925 Train 0.844 Random State 152
Test 0.850 Train 0.841 Random State 153
Test 0.900 Train 0.844 Random State 154
Test 0.900 Train 0.841 Random State 155
Test 0.887 Train 0.847 Random State 156
Test 0.887 Train 0.834 Random State 158
Test 0.875 Train 0.828 Random State 159
Test 0.900 Train 0.831 Random State 161

Test 0.850 Train 0.838 Random State 163
Test 0.875 Train 0.831 Random State 164
Test 0.863 Train 0.850 Random State 169
Test 0.875 Train 0.841 Random State 171
Test 0.850 Train 0.841 Random State 172
Test 0.900 Train 0.825 Random State 180
Test 0.850 Train 0.834 Random State 184
Test 0.925 Train 0.822 Random State 186
Test 0.900 Train 0.831 Random State 193
Test 0.863 Train 0.850 Random State 195
Test 0.863 Train 0.841 Random State 196
Test 0.863 Train 0.838 Random State 197
Test 0.875 Train 0.841 Random State 198
Test 0.887 Train 0.838 Random State 199
Test 0.887 Train 0.844 Random State 200
Test 0.863 Train 0.838 Random State 202
Test 0.863 Train 0.841 Random State 203
Test 0.887 Train 0.831 Random State 206
Test 0.863 Train 0.834 Random State 211
Test 0.850 Train 0.844 Random State 212
Test 0.863 Train 0.834 Random State 214
Test 0.875 Train 0.831 Random State 217
Test 0.963 Train 0.819 Random State 220
Test 0.875 Train 0.844 Random State 221
Test 0.850 Train 0.841 Random State 222
Test 0.900 Train 0.844 Random State 223
Test 0.863 Train 0.853 Random State 227
Test 0.863 Train 0.834 Random State 228
Test 0.900 Train 0.841 Random State 229
Test 0.850 Train 0.844 Random State 232
Test 0.875 Train 0.847 Random State 233
Test 0.912 Train 0.841 Random State 234
Test 0.863 Train 0.841 Random State 235
Test 0.850 Train 0.847 Random State 236
Test 0.875 Train 0.847 Random State 239
Test 0.850 Train 0.844 Random State 241
Test 0.887 Train 0.850 Random State 242
Test 0.887 Train 0.825 Random State 243
Test 0.875 Train 0.847 Random State 244
Test 0.875 Train 0.841 Random State 245
Test 0.875 Train 0.847 Random State 246
Test 0.863 Train 0.859 Random State 247
Test 0.887 Train 0.844 Random State 248
Test 0.863 Train 0.850 Random State 250
Test 0.875 Train 0.831 Random State 251
Test 0.887 Train 0.844 Random State 252
Test 0.863 Train 0.847 Random State 255
Test 0.900 Train 0.841 Random State 257

Test 0.863 Train 0.856 Random State 260
Test 0.863 Train 0.841 Random State 266
Test 0.863 Train 0.838 Random State 268
Test 0.875 Train 0.841 Random State 275
Test 0.863 Train 0.850 Random State 276
Test 0.925 Train 0.838 Random State 277
Test 0.875 Train 0.847 Random State 282
Test 0.850 Train 0.847 Random State 283
Test 0.850 Train 0.844 Random State 285
Test 0.912 Train 0.834 Random State 286
Test 0.850 Train 0.841 Random State 290
Test 0.850 Train 0.841 Random State 291
Test 0.850 Train 0.847 Random State 292
Test 0.863 Train 0.838 Random State 294
Test 0.887 Train 0.828 Random State 297
Test 0.863 Train 0.834 Random State 300
Test 0.863 Train 0.850 Random State 301
Test 0.887 Train 0.850 Random State 302
Test 0.875 Train 0.847 Random State 303
Test 0.863 Train 0.834 Random State 305
Test 0.912 Train 0.838 Random State 306
Test 0.875 Train 0.847 Random State 308
Test 0.900 Train 0.844 Random State 311
Test 0.863 Train 0.834 Random State 313
Test 0.912 Train 0.834 Random State 314
Test 0.875 Train 0.838 Random State 315
Test 0.900 Train 0.847 Random State 317
Test 0.912 Train 0.822 Random State 319
Test 0.863 Train 0.850 Random State 321
Test 0.912 Train 0.828 Random State 322
Test 0.850 Train 0.847 Random State 328
Test 0.850 Train 0.838 Random State 332
Test 0.887 Train 0.853 Random State 336
Test 0.850 Train 0.838 Random State 337
Test 0.875 Train 0.841 Random State 343
Test 0.863 Train 0.844 Random State 346
Test 0.887 Train 0.831 Random State 351
Test 0.863 Train 0.850 Random State 352
Test 0.950 Train 0.819 Random State 354
Test 0.863 Train 0.850 Random State 356
Test 0.912 Train 0.841 Random State 357
Test 0.863 Train 0.838 Random State 358
Test 0.850 Train 0.841 Random State 362
Test 0.900 Train 0.844 Random State 363
Test 0.863 Train 0.853 Random State 364
Test 0.938 Train 0.822 Random State 366
Test 0.912 Train 0.841 Random State 369
Test 0.863 Train 0.853 Random State 371

```

Test 0.925 Train 0.834 Random State 376
Test 0.912 Train 0.828 Random State 377
Test 0.887 Train 0.850 Random State 378
Test 0.887 Train 0.850 Random State 379
Test 0.863 Train 0.841 Random State 382
Test 0.863 Train 0.859 Random State 386
Test 0.850 Train 0.838 Random State 387
Test 0.875 Train 0.828 Random State 388
Test 0.850 Train 0.844 Random State 394
Test 0.863 Train 0.838 Random State 395
Test 0.900 Train 0.844 Random State 397
Test 0.863 Train 0.844 Random State 400

```

```

[7]: x_train, x_test, y_train, y_test = train_test_split(features, label,
↳test_size=0.2, random_state=42)
finalModel = LogisticRegression()
finalModel.fit(x_train, y_train)

```

```

[7]: LogisticRegression()

```

```

[8]: print(finalModel.score(x_train,y_train))
print(finalModel.score(x_test,y_test))

```

```

0.8375
0.8875

```

```

[9]: from sklearn.metrics import classification_report
print(classification_report(label,finalModel.predict(features)))

```

	precision	recall	f1-score	support
0	0.85	0.93	0.89	257
1	0.85	0.70	0.77	143
accuracy			0.85	400
macro avg	0.85	0.81	0.83	400
weighted avg	0.85	0.85	0.84	400

```

[ ]:

```