Supervised Learning: Classification

3 min

Now that you've seen a regression example, let's take a look at a classification example.

An exclusive nightclub in Neo York doesn't want to serve robots, but technology has advanced so far that it's hard for bouncers to tell humans from robots just by looking. To help the bouncers, the nightclub created a model that uses the k-nearest neighbors algorithm to distinguish between humans and robots based on how long it takes them identify blurry pictures or traffic lights.

Instructions

1. Checkpoint 1 Passed

1.

Run the code to train the model and see whether someone who identifies a picture in 5 seconds can pass as a human.

Stuck? Get extra guidance

2. Checkpoint 2 Passed

2.

Change the variable time_to_identify_picture to a different value. Run the code again. Does the model still think that you're a human?

If you want to experiment more you can change the value again and see how that affects the outcome.

script.py

```
import numpy as np

import pandas as pd

from sklearn.neighbors import KNeighborsClassifier


# Load the data

photo_id_times = pd.read_csv('photo_id_times.csv')


# Separate the data into independent and dependent variables

X = np.array(photo_id_times['Time to id photo']).reshape(-1, 1)

y = photo_id_times['Class']


# Create a model and fit it to the data

neigh = KNeighborsClassifier(n_neighbors=3)

neigh.fit(X, y)
```

```
time_to_identify_picture = 5

# Make a prediction based on how long it takes to identify a picture
y_pred = neigh.predict(np.array(time_to_identify_picture).reshape(1, -1))

if y_pred == 1:
   print("We think you're a robot.")
else:
   print("Welcome, human!")
```

photo_id_times.csv

```
Time to id photo,Class
3.538784262035922,0.0
1.6741631218885176,0.0
2.6970051251131326,0.0
0.50616052794562,0.0
2.344699699619277,0.0
1.3421530626411964,0.0
1.3893504538451886,0.0
3.979478785176629,0.0
3.7792664032257153,0.0
4.594921533518095,0.0
2.5057227935336073,0.0
2.9285649394441937,0.0
1.5263189902675653,0.0
-0.31170910669813656,0.0
4.686282773865226,0.0
3.2305113005884523,0.0
4.581573620088822,0.0
3.2730377192285762,0.0
5.828922897421932,0.0
3.2086210806708695,0.0
```

3.010915341252376,0.0

2.627243672018273,0.0

5.408309295801245,0.0

3.64817649946776,0.0

2.033852899664028,0.0

5.085773402364357,0.0

3.731915215672551,0.0

4.738513385381877,0.0

3.100048875213451,0.0

4.646820275504821,0.0

3.6214778131003533,0.0

1.5334591714222052,0.0

3.3705428158625157,0.0

4.4075344385486055,0.0

3.0294071947850147,0.0

3.4019895108854845,0.0

3.0665617648308117,0.0

3.306263562486243,0.0

5.3067957573356015,0.0

3.862067934979338,0.0

1.5507657998313948,0.0

5.02693253753233,0.0

5.0496722817352415,0.0

2.571291483808387,0.0

6.0247360729816855,0.0

4.087801394510661,0.0

0.007052842548844751,0.0

3.64441021191731,0.0

1.6406462639771062,0.0

2.6508279014215006,0.0

2.4016334002212947,0.0

4.680555212313692,0.0

1.3244041394513268,0.0

3.3727282400297596,0.0

2.875589854287772,0.0

1.1366238288740353,0.0

2.904277403776058,0.0

0.6133799465024801,0.0

-1.0432706266455778,0.0

1.220746490517254,0.0

7.853800938004747,1.0

6.6710583006770765,1.0

6.840138862102425,1.0

7.034293541861588,1.0

7.969819882962885,1.0

5.581016006813048,1.0

3.5125249286777303,1.0

5.991214259837085,1.0

6.372915371772184,1.0

10.00734801810078,1.0

6.086478260867031,1.0

6.848469122077413,1.0

4.887062856222357,1.0

7.69947906723694,1.0

6.394511406288887,1.0

8.379354708805439,1.0

5.721855327740236,1.0

9.092083350083113,1.0

5.666081377644842,1.0

9.480739094135927,1.0

7.519806083236695,1.0

7.504158600437411,1.0

7.222527216905084,1.0

9.219363018813269,1.0

7.61183563409749,1.0

7.521638961771235,1.0

7.481794859933867,1.0

8.351710177529444,1.0

6.518514666368797,1.0

5.162924328955024,1.0

7.0520638389247,1.0

6.868049753878597,1.0

7.452312302584311,1.0

7.743559633647344,1.0

5.300488885813206,1.0

8.221441305701802,1.0

5.25666155498552,1.0

8.059844235600853,1.0

3.8873423773267857,1.0

5.086033337983446,1.0