
Cascade Classify

Narendra Arkan Putra Darmawan

1313621043

Muhammad Ramadhan Putra Pratama

1313621038

1. Fungsi calculate_sumless_mean(data)

```
1  function calculate_sumless_mean(data)
2      averages = Matrix{Float16}(undef, 3, 4)
3
4      for (i, val) in enumerate(1:3)
5          filtered_data = data[data[:, 5] .== val, :]
6
7          for col in 1:4
8              n = 0
9              mean = 0.0
10
11              for x in filtered_data[:, col]
12                  n += 1
13                  delta = x - mean
14                  mean += delta / n
15              end
16
17              averages[i, col] = mean
18          end
19      end
20      return averages
21  end
```

- Fungsi ini menghitung rata-rata setiap fitur (kolom) untuk setiap kelas (nilai 1 hingga 3).
- Variabel averages adalah matriks yang menyimpan hasil perhitungan rata-rata.
- Data difilter berdasarkan kelas, dan rata-rata dihitung dengan metode Welford untuk menghindari sensitivitas terhadap urutan data.

2. Fungsi predict_class(data, averages)

```
1 function predict_class(data, averages)
2     predicted_classes = Matrix{Float16}(undef, size(data, 1), 4)
3
4     for row in axes(data, 1)
5
6         for col in axes(data, 2)
7             distances = zeros(Float16, 3)
8
9             for i in 1:3
10                 distances[i] = abs(data[row, col] - averages[i, col])
11             end
12
13             predicted_classes[row, col] = argmin(distances)
14         end
15     end
16
17     return predicted_classes
18 end
```

- Fungsi ini memprediksi kelas untuk setiap data dalam dataset.
- Menggunakan algoritma Euclidean dengan membandingkan jarak antara nilai fitur dan rata-rata yang telah dihitung.
- Hasil prediksi disimpan dalam matriks predicted_classes.

3. Fungsi `compare_classes(predicted_classes, actual_classes)`

```
1  function compare_classes(predicted_classes, actual_classes)
2      comparison = Matrix{Bool}(undef, size(predicted_classes, 1), 4)
3
4      for row in axes(predicted_classes, 1)
5
6          for col in axes(predicted_classes, 2)
7              comparison[row, col] = predicted_classes[row, col] == actual_classes[row]
8          end
9      end
10
11     return comparison
12 end
```

- Fungsi ini membandingkan hasil prediksi kelas dengan kelas sebenarnya.
- Membuat matriks comparison yang berisi nilai boolean apakah prediksi benar atau salah.

4. Fungsi calculate_accuracy(comparison)

```
1  function calculate_accuracy(comparison)
2      accuracy = zeros(Float32, 4)
3
4      for i in 1:4
5          accuracy[i] = mean(comparison[:, i])
6      end
7
8      return accuracy
9  end
10
```

- Fungsi ini menghitung akurasi untuk setiap fitur berdasarkan matriks perbandingan.
- Menghasilkan nilai akurasi untuk masing-masing fitur.

5. Fungsi filter_correct_prediction(data, comparison)

```
1 function filter_correct_prediction(data, comparison, accuracies)
2     correct_size = 0
3     incorrect_size = 0
4     for row in axes(data, 1)
5         if comparison[row, argmax(accuracies)] == true
6             correct_size += 1
7         else
8             incorrect_size += 1
9         end
10    end
11
12    correct_data = Matrix{Float16}(undef, correct_size, 5)
13    incorrect_data = Matrix{Float16}(undef, incorrect_size, 5)
14    incorrect_comparison = zeros{Bool}(incorrect_size, 4)
15    i = 1
16    j = 1
17    for row in axes(data, 1)
18        if comparison[row, argmax(accuracies)] == true
19            for col in 1:5
20                correct_data[i, col] = data[row, col]
21            end
22            i += 1
23        else
24            for col in 1:5
25                incorrect_data[j, col] = data[row, col]
26                incorrect_comparison[j, :] = comparison[row, :]
27            end
28            j += 1
29        end
30    end
31
32    return correct_data, incorrect_data, incorrect_comparison
33 end
```

- Fungsi ini memisahkan data yang diprediksi dengan benar dan salah.
- Mengembalikan dua matriks terpisah: correct_data dan incorrect_data

6. Fungsi correct_prediction_sumless_mean(data, col)

```
1  function correct_prediction_sumless_mean(data, col)
2      averages = Matrix{Float16}(undef, 3, 1)
3
4      for (i, val) in enumerate(1:3)
5          filtered_data = data[data[:, 5] .== val, :]
6          n = 0
7          mean = 0.0
8
9          for x in filtered_data[:, col]
10              n += 1
11              delta = x - mean
12              mean += delta / n
13          end
14
15          averages[i, 1] = mean
16      end
17
18      return averages
19 end
20
```

- Fungsi ini menghitung rata-rata untuk suatu fitur dari data yang diprediksi dengan benar.
- Digunakan untuk menghasilkan rata-rata baru berdasarkan data yang diprediksi dengan benar.

7. Fungsi calculate_correct_prediction_averages(data, comparison, accuracies)

```
1 function calculate_correct_prediction_averages(data, comparison, accuracies)
2     averages = Matrix{Float16}(undef, 3, 4)
3     temp_accuracies = copy(accuracies)
4
5     x1, remainder_data, remainder_comparison = filter_correct_prediction(data, comparison, temp_accuracies)
6     x1_averages = correct_prediction_sumless_mean(x1, argmax(temp_accuracies))
7     averages[:, argmax(temp_accuracies)] = x1_averages
8     temp_accuracies[argmax(temp_accuracies)] = 0.0
9
10    x2, remainder_data, remainder_comparison = filter_correct_prediction(remainder_data, remainder_comparison, temp_accuracies)
11    x2_averages = correct_prediction_sumless_mean(x2, argmax(temp_accuracies))
12    averages[:, argmax(temp_accuracies)] = x2_averages
13    temp_accuracies[argmax(temp_accuracies)] = 0.0
14
15    x3, remainder_data, remainder_comparison = filter_correct_prediction(remainder_data, remainder_comparison, temp_accuracies)
16    x3_averages = correct_prediction_sumless_mean(x3, argmax(temp_accuracies))
17    averages[:, argmax(temp_accuracies)] = x3_averages
18    temp_accuracies[argmax(temp_accuracies)] = 0.0
19
20    x4, remainder_data, remainder_comparison = filter_correct_prediction(remainder_data, remainder_comparison, temp_accuracies)
21    x4_averages = correct_prediction_sumless_mean(x4, argmax(temp_accuracies))
22    averages[:, argmax(temp_accuracies)] = x4_averages
23    temp_accuracies[argmax(temp_accuracies)] = 0.0
24
25    return averages
26 end
```

- Fungsi ini menghasilkan rata-rata baru berdasarkan data yang diprediksi dengan benar.
- Menggunakan 2 fungsi sebelumnya untuk memfilter data dan menghitung rata-rata berdasarkan fitur.
- Digunakan untuk pembaruan averages setelah iterasi pertama klasifikasi.

8. Pemanggilan Fungsi dan Eksekusi

```
1 raw_data = deserialize(open("data_9m.mat", "r"))
2 println("Raw Data: ")
3 display(raw_data)
4
5 averages = calculate_sumless_mean(raw_data)
6 println("\nAverages: ")
7 display(averages)
8
9 predicted_classes = predict_class(raw_data[:, 1:4], averages)
10 println("\nPredicted Classes:")
11 display(predicted_classes)
12
13 comparison = compare_classes(predicted_classes, raw_data[:, 5])
14 println("\nCorrect Predictions: ")
15 display(comparison)
16
17 accuracies = calculate_accuracy(comparison)
18 println("\nFeature 1 Accuracy: ${(accuracies[1] * 100)%}")
19 println("Feature 2 Accuracy: ${(accuracies[2] * 100)%}")
20 println("Feature 3 Accuracy: ${(accuracies[3] * 100)%}")
21 println("Feature 4 Accuracy: ${(accuracies[4] * 100)%}")
22
23 correct_prediction_averages = calculate_correct_prediction_averages(raw_data, comparison, accuracies)
24 println("\nNew Averages: ")
25 display(correct_prediction_averages)
```

- Data yang dideserialize dari file "data_9m.mat" dibaca dan digunakan dalam analisis.
- Setelah melakukan klasifikasi, averages baru dihitung berdasarkan data yang diprediksi dengan benar

9. Output Program

```
Raw Data:
9830400x5 Matrix{Float16}:
  5.1    3.5    1.4    0.2    1.0
  4.9    3.0    1.4    0.2    1.0
  4.7    3.2    1.3    0.2    1.0
  4.6    3.1    1.5    0.2    1.0
  ⋮
 25.25  -1.852 -7.69  -11.695  3.0
  3.094   1.225 -6.336  13.93   3.0
 12.44  -10.96 -14.85  -13.12   3.0
 -1.461 -10.664 24.47  -10.44   3.0
```

```
Averages:
3x4 Matrix{Float16}:
 3.229  1.777  0.541  -2.166
 4.492  1.169  3.797  -0.517
 5.754  0.92   3.963  -0.3533
```

```
Predicted Classes:
9830400x4 Matrix{Float16}:
 2.0  1.0  1.0  3.0
 2.0  1.0  1.0  3.0
 2.0  1.0  1.0  3.0
 2.0  1.0  1.0  3.0
  ⋮
 3.0  3.0  1.0  1.0
 1.0  2.0  1.0  3.0
 3.0  3.0  1.0  1.0
 1.0  3.0  3.0  1.0
```

- Hasil deserialize data mentah
- Rata-rata atau μ (mu) dari tiap features dan class
- Hasil prediksi class berdasarkan euclidean distance dari tiap features

10. Output Program

```
Correct Predictions:
9830400x4 Matrix{Bool}:
0 1 1 0
0 1 1 0
0 1 1 0
0 1 1 0
⋮
1 1 0 0
0 0 0 1
1 1 0 0
0 1 1 0
```

- Perbandingan predicted classes dengan class asli. Jika sama maka nilai = 1 (true), jika berbeda maka nilai = 0 (false)

```
Feature 1 Accuracy: 39.67047%
Feature 2 Accuracy: 36.10197%
Feature 3 Accuracy: 44.05227%
Feature 4 Accuracy: 40.380157%
```

- Akurasi dari prediksi berdasarkan euclidean distance dari tiap feature

```
New Averages:
3x4 Matrix{Float16}:
-1.096  6.125  -2.451  -5.52
 4.516  1.264  3.059  -0.861
10.55  -2.803  8.53   4.105
```

- Rata-rata baru diambil data yang hasil prediksinya benar