Pert 9 - Shiny

YOHANES FEBRYAN KANA NYOLA_123220198

2024-11-19

Regresi Logistik dan Shiny App

Import Library

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```
library(tidymodels)
```

```
## ── Attaching packages ─────────────────────────────────────── tidymodels 1.2.0 ──
## ✓ broom        1.0.6     ✓ rsample      1.2.1
## ✓ dials        1.3.0     ✓ tune         1.2.1
## ✓ infer        1.0.7     ✓ workflows    1.1.4
## ✓ modeldata    1.4.0     ✓ workflowsets 1.1.0
## ✓ parsnip      1.2.1     ✓ yardstick    1.3.1
## ✓ recipes      1.1.0
## ── Conflicts ────────────────────────────────────── tidymodels_conflicts() ──
## ✖ scales::discard() masks purrr::discard()
## ✖ dplyr::filter()   masks stats::filter()
## ✖ recipes::fixed()  masks stringr::fixed()
## ✖ dplyr::lag()      masks stats::lag()
## ✖ yardstick::spec() masks readr::spec()
## ✖ recipes::step()   masks stats::step()
## • Use tidymodels_prefer() to resolve common conflicts.
```

```
library(nnet) # Model multinomial logistic regression

library(shiny) # Aplikasi
```

```
## Warning: package 'shiny' was built under R version 4.4.2
```

```
##
## Attaching package: 'shiny'
##
## The following object is masked from 'package:infer':
##
##     observe
```

```
library(bslib)
```

```
##
## Attaching package: 'bslib'
##
## The following object is masked from 'package:broom':
##
##     bootstrap
##
## The following object is masked from 'package:utils':
##
##     page
```

```
library(datasets) # Dataset iris
```

Import Data

```
data(iris)
# View(iris)
```

Regresi Logistik

Inisialisasi Kelas

```
kelas = levels(iris$Species)
kelas
```

```
## [1] "setosa"     "versicolor" "virginica"
```

Data Splitting

```
# Set seed untuk memastikan hasil pembagian data yang sama setiap kali dijalankan
set.seed(420)

# Membagi dataset Iris menjadi 80% data latih (training) dan 20% data uji (testing), dengan s
tratifikasi berdasarkan kolom Species
split = initial_split(iris, prop = 0.8, strata = Species)

# Membuat dataset latih dari pembagian data
iris_train = split %>% training()

# Membuat dataset uji dari pembagian data
iris_test = split %>% testing()

# Meringkas data latih berdasarkan spesies, menghitung frekuensi setiap spesies
iris_train %>%
  select(Species) %>% # Memilih kolom Species
  group_by(Species) %>% # Mengelompokkan data berdasarkan Species
  summary(freq=n()) # Meringkas jumlah data per spesies
```

```
##        Species
##  setosa    :40
##  versicolor:40
##  virginica :40
```

```
# Meringkas data uji berdasarkan spesies, menghitung frekuensi setiap spesies
iris_test %>%
  select(Species) %>% # Memilih kolom Species
  group_by(Species) %>% # Mengelompokkan data berdasarkan Species
  summary(freq=n()) # Meringkas jumlah data per spesies
```

```
##        Species
##  setosa    :10
##  versicolor:10
##  virginica :10
```

Modelling

```
# Melatih model multinomial logistic regression menggunakan dataset latih
hasil_model = multinom(
  Species ~ ., # Species adalah variabel target, sedangkan "." merepresentasikan semua fitur
lain sebagai prediktor
  data = iris_train # Data latih yang digunakan untuk melatih model
)
```

```
## # weights:  18 (10 variable)
## initial  value 131.833475
## iter  10 value 11.748662
## iter  20 value 4.655068
## iter  30 value 4.358366
## iter  40 value 4.001262
## iter  50 value 3.994115
## iter  60 value 3.988789
## iter  70 value 3.988023
## iter  80 value 3.987341
## iter  90 value 3.987049
## iter 100 value 3.986969
## final  value 3.986969
## stopped after 100 iterations
```

```
# Menampilkan ringkasan model multinomial logistic regression
summary(hasil_model)
```

```
## Call:
## multinom(formula = Species ~ ., data = iris_train)
##
## Coefficients:
##            (Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width
## versicolor    18.80645    -5.894053   -7.504432     13.10109   -1.330924
## virginica    -22.83089    -7.415324  -10.873479     20.16393   14.975692
##
## Std. Errors:
##            (Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width
## versicolor    32.90631     80.07097    135.5134     55.08394    38.61356
## virginica     33.75244     80.06341    135.5774     55.26723    38.98935
##
## Residual Deviance: 7.973938
## AIC: 27.97394
```

Data Testing

```
# type = "probs", outputnya berupa probabilitas tiap kelas

# Membuat prediksi probabilitas untuk data uji menggunakan model yang telah dilatih
hasil_prediksi_probs = predict(
  hasil_model, # Model multinomial logistic regression yang telah dilatih
  newdata = iris_test, # Data uji yang digunakan untuk prediksi
  type = "probs" # Menghasilkan probabilitas untuk setiap kelas (bukan prediksi kelas langsun
g)
)

# Mengonversi probabilitas menjadi persentase dan membulatkan hingga 2 desimal
hasil_prediksi_probs = round(hasil_prediksi_probs * 100 , digits = 2)

# Menampilkan hasil prediksi probabilitas dalam bentuk tabel
hasil_prediksi_probs
```

```
##    setosa versicolor virginica
## 1     100       0.00      0.00
## 2     100       0.00      0.00
## 3     100       0.00      0.00
## 4     100       0.00      0.00
## 5     100       0.00      0.00
## 6     100       0.00      0.00
## 7     100       0.00      0.00
## 8     100       0.00      0.00
## 9     100       0.00      0.00
## 10    100       0.00      0.00
## 11      0      95.49      4.51
## 12      0      99.98      0.02
## 13      0      13.47     86.53
## 14      0     100.00      0.00
## 15      0      64.02     35.98
## 16      0      25.00     75.00
## 17      0     100.00      0.00
## 18      0     100.00      0.00
## 19      0      97.63      2.37
## 20      0      99.67      0.33
## 21      0       0.04     99.96
## 22      0       0.00    100.00
## 23      0       0.00    100.00
## 24      0       0.14     99.86
## 25      0       0.00    100.00
## 26      0       1.26     98.74
## 27      0       0.00    100.00
## 28      0       0.01     99.99
## 29      0       0.13     99.87
## 30      0       0.04     99.96
```

```
# type = "class", outputnya berupa kelas

# Membuat prediksi kelas untuk data uji menggunakan model multinomial logistic regression
hasil_prediksi_class = predict(
  hasil_model, # Model yang telah dilatih sebelumnya
  newdata = iris_test, # Dataset uji yang akan diprediksi
  type = "class" # Parameter "class" mengembalikan prediksi kelas langsung
)

# Menampilkan hasil prediksi kelas
hasil_prediksi_class
```

```
##  [1] setosa     setosa     setosa     setosa     setosa     setosa
##  [7] setosa     setosa     setosa     setosa     versicolor versicolor
## [13] virginica  versicolor versicolor virginica  versicolor versicolor
## [19] versicolor versicolor virginica  virginica  virginica  virginica
## [25] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

Evaluasi Model

```
# Membuat tabel perbandingan antara kelas yang diprediksi dan kelas sebenarnya
table(
  predicted_class = hasil_prediksi_class, # Kelas yang diprediksi oleh model
  actual_class = iris_test$Species # Kelas sebenarnya dari data uji
)
```

```
##                actual_class
## predicted_class setosa versicolor virginica
##      setosa         10          0         0
##      versicolor      0          8         0
##      virginica       0          2        10
```

```
# Menghitung akurasi model secara manual
akurasi = (10 + 8 + 10) / (10 + 8 + 2 + 10) * 100 # (Jumlah prediksi benar) / (Total prediks
i) * 100

# Menampilkan nilai akurasi
akurasi
```

```
## [1] 93.33333
```

```
# Membuat data frame baru
data_frame = data.frame(
  predicted_class = hasil_prediksi_class, # Kolom pertama: hasil prediksi
  actual_class = iris_test$Species # Kolom kedua: kelas aktual dari data test
)

nrow(data_frame %>% # Hitung jumlah baris dari data frame
       filter(predicted_class == actual_class)) / # Filter baris di mana prediksi sama dengan
kelas aktual
  nrow(data_frame) * 100 # Bagi dengan jumlah total baris, lalu dikalikan 100 untuk menghitun
g akurasi
```

```
## [1] 93.33333
```

Shiny App

Membuat UI

```
ui = fluidPage(
  # Tambahkan CSS untuk tata letak tengah
  tags$style(HTML("
    .center {
      display: flex;
      flex-direction: column;
      align-items: center;
      justify-content: center;
    }
    table {
      margin-top: 20px;
      border-collapse: collapse;
    }
    .main-title {
      text-align: center;
      margin-bottom: 20px;
    }
  ")),

  # Judul utama di luar div "center"
  div(
    class = "main-title",
    titlePanel("Dataset Iris")
  ),

  # Wrapper div untuk elemen di tengah
  div(
    class = "center",

    # Dropdown untuk memilih spesies
    selectInput(
      inputId = "species",
      label = "Pilih Jenis Spesies : ",
      choices = kelas
    ),

    # Output tabel
    tableOutput(outputId = "table_iris"),

    # Jarak tambahan
    tags$br(),

    # Judul kedua
    titlePanel("Uji Coba"),

    # Layout untuk input kolom
    fluidRow(
      column(3, numericInput(inputId = "sl", label = "Sepal Length", value = 1)),
      column(3, numericInput(inputId = "sw", label = "Sepal Width", value = 1)),
      column(3, numericInput(inputId = "pl", label = "Petal Length", value = 1)),
      column(3, numericInput(inputId = "pw", label = "Petal Width", value = 1))
```

```
    ),

    # Tombol klasifikasi
    actionButton(
      inputId = "klasifikasi",
      label = "Klasifikasi"
    ),

    # Jarak tambahan
    tags$br(),
    tags$br(),

    # Output hasil klasifikasi
    textOutput(outputId = "hasil_klasifikasi"),

    # Jarak tambahan
    tags$br()
  )
)
```

Membuat Logika Di Belakang Layar

```
server = function(input, output){ # Definisi fungsi server untuk Shiny
  output$table_iris = renderTable( # Membuat output tabel untuk menampilkan data iris
    head( # Menampilkan 10 baris pertama
      iris %>%  # Data iris
        filter( # Filter data sesuai spesies
          Species == input$species # Kondisi filter: spesies sesuai input dari pengguna
        ),
      10 # Ambil 10 baris pertama
    )
  )

  output$hasil_klasifikasi = renderText({ # Membuat output berupa teks hasil klasifikasi
    input_prediksi = data.frame( # Membuat data frame baru untuk prediksi
      Sepal.Length = input$sl, # Kolom panjang sepal dari input pengguna
      Sepal.Width = input$sw, # Kolom lebar sepal dari input pengguna
      Petal.Length = input$pl, # Kolom panjang petal dari input pengguna
      Petal.Width = input$pw # Kolom lebar petal dari input pengguna
    )

    hasil_class = predict( # Melakukan prediksi menggunakan model
      hasil_model, # Model yang telah dilatih
      newdata = input_prediksi, # Data input untuk prediksi
      type = "class" # Prediksi berupa kelas (label)
    )

    nama_kelas = kelas[hasil_class] # Mengambil nama kelas dari hasil prediksi

    hasil_probs = predict( # Menghitung probabilitas dari prediksi
      hasil_model, # Model yang telah dilatih
      newdata = input_prediksi, # Data input untuk prediksi
      type = "probs" # Prediksi berupa probabilitas
    )

    persentase = round( # Menghitung probabilitas tertinggi dan membulatkan
      max(hasil_probs) * 100, # Ambil probabilitas maksimum dan konversi ke persen
      digits = 2 # Membulatkan hingga 2 angka desimal
    )

    paste( # Menggabungkan nama kelas dan persentase ke dalam string
      nama_kelas, # Nama kelas hasil prediksi
      " (",
      persentase, # Persentase prediksi
      "%)",
      sep = "" # Menghilangkan spasi tambahan
    )
  })|> bindEvent(input$klasifikasi) # Render output hanya ketika tombol klasifikasi diklik
}
```

Run Aplikasi

```
shinyApp(ui,server) # Menjalankan aplikasi Shiny dengan antarmuka pengguna (UI) dan fungsi se
rver
```

Shiny applications not supported in static R Markdown documents