2. Fish Classifier

Gruppe 23 Justin König, Ali Abdullah, Rita Tagoula Ngoufo

10.05.2023

Contents

Daten

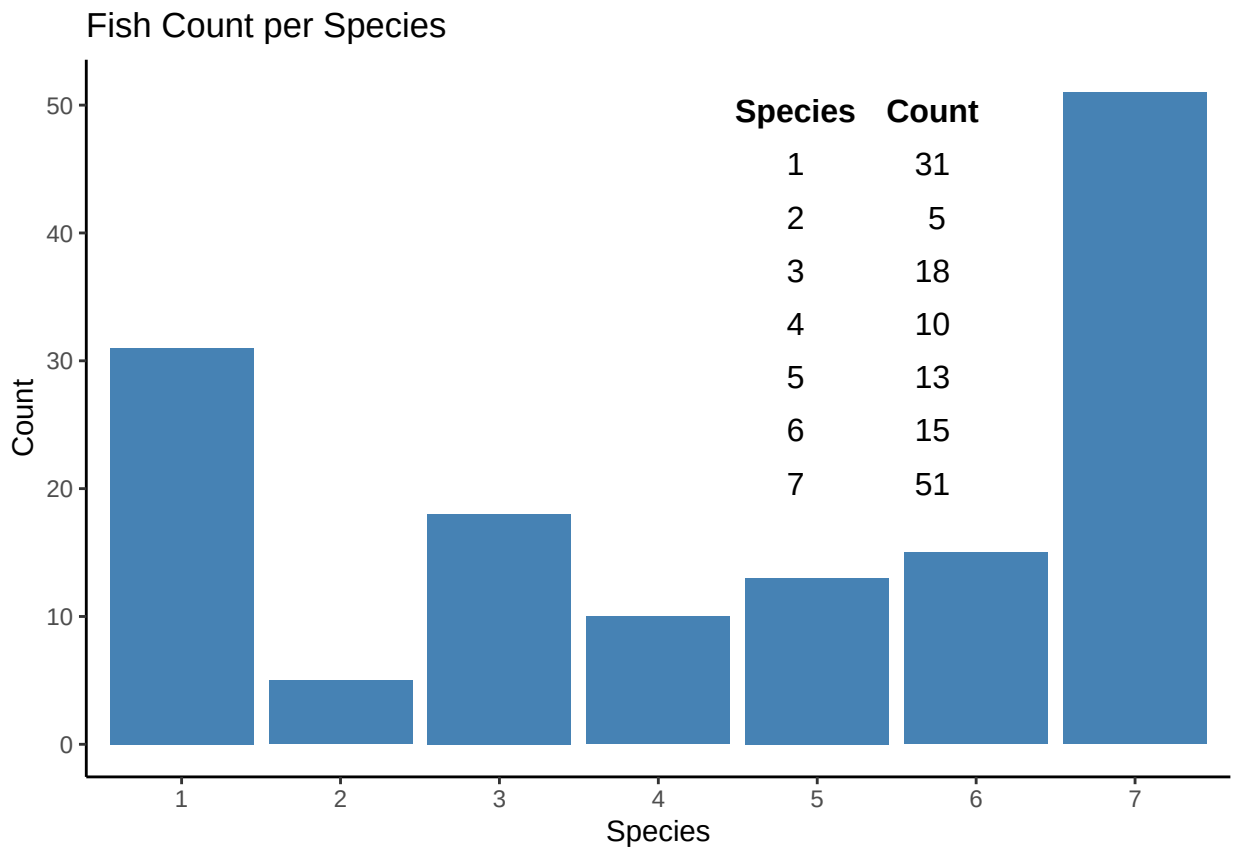
Ein Datensatz, der aus Länge1, Länge2, Länge3, Höhe, Breite, Geschlecht und Gewicht von 7 verschiedenen Fischarten besteht, wurde für die Analyse verwendet. Ziel war es, einen Klassifikator zu entwickeln, der die Fischspezies basierend auf den Merkmalen Länge1, Länge2, Länge3, Höhe und Breite in einem Testdatensatz vorhersagen kann. Der Testdatensatz wurde aus jeder 10. Zeile des ursprünglichen Datensatzes gebildet. Die verbleibenden Daten wurden als Trainingsdatensatz verwendet, um den Klassifikator zu trainieren und dessen Leistung anhand des Testdatensatzes zu bewerten.

```
# load data frame
fish_df <- read.csv2("fishcatch.csv")

# remove unnecessary data
fish_df <- fish_df[48:nrow(fish_df),]

colnames(fish_df) <- c("Species", "Length1", "Length2", "Length3","Height",
                       "Width", "sex", "Weight")

fish_df <- as.data.frame(apply(fish_df, 2, function(x) gsub(",", ".", x)))

# List the columns you want to convert to numeric
columns_to_convert <- c("Length1", "Length2", "Length3","Height", "Width",
                       "Weight")

# Convert the specified columns to numeric
fish_df <- fish_df %>%
```

```
  mutate(across(all_of(columns_to_convert), as.numeric))

# training data (data frame without every 10th row)
train_data <- fish_df[!(seq_len(nrow(fish_df)) %% 10 == 0), ]

# test df (every 10. Fish)
test_data <- fish_df[seq(1, nrow(fish_df), by = 10), ]

# counts fish classes
fish_count <- train_data %>%
  group_by(Species) %>%
  summarise(Count = n())

# creates a tibble out of the df
fish_count <- as_tibble(fish_count)
```

Im folgenden Diagramm ist die Anzahl der jeweiligen Fischspezies im Trainingsatensatz dargstellt.

```
# creates the theme tt2 with minimal settings
tt2 <- ttheme_minimal()

# creates a grob to put inside of the bar plot
fish_count_grob <- tableGrob(fish_count, theme = tt2, rows = NULL)

fish_count_plot <- ggplot(fish_count, aes(x = Species, y = Count)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  theme_classic() +
  labs(title = "Fish Count per Species",
       x = "Species",
       y = "Count")

fish_count_plot_with_table <- fish_count_plot +
  annotation_custom(
    grob = fish_count_grob,
    xmin = 5, xmax = 5.5, ymin = 20, ymax = 50
  )
print(fish_count_plot_with_table)
```



Species	Count
1	31
2	5
3	18
4	10
5	13
6	15
7	51

Die Normalverteilungsparameter sind in der folgender Tabelle zusammengefasst.

```
# group by species
grouped_fish <- fish_df %>%
  group_by(Species)

# calculates mean and standarddeviation
normal_params <- grouped_fish %>%
  summarise(
    "mean L1" = mean(Length1, na.rm = TRUE),
    "mean L2" = mean(Length2, na.rm = TRUE),
    "mean L3" = mean(Length3, na.rm = TRUE),
    "mean H" = mean(Height, na.rm = TRUE),
    "mean W" = mean(Width, na.rm = TRUE),
    "SD L1" = sd(Length1, na.rm = TRUE),
    "SD L2" = sd(Length2, na.rm = TRUE),
    "SD L3" = sd(Length3, na.rm = TRUE),
    "SD H" = sd(Height, na.rm = TRUE),
    "SD W" = sd(Width, na.rm = TRUE),
  )

rounded_normal_params <- normal_params %>%
  mutate_if(is.numeric, round, digits = 2)

# creates a table out of the tibble
normal_params_gt <- gt(rounded_normal_params)
```

```r
# changes to the table
normal_params_gt <-
  normal_params_gt %>%
  # Titel
  tab_header(
    title = "Normal distribution parameters ",
  )

normal_params_gt
```

Normal distribution parameters

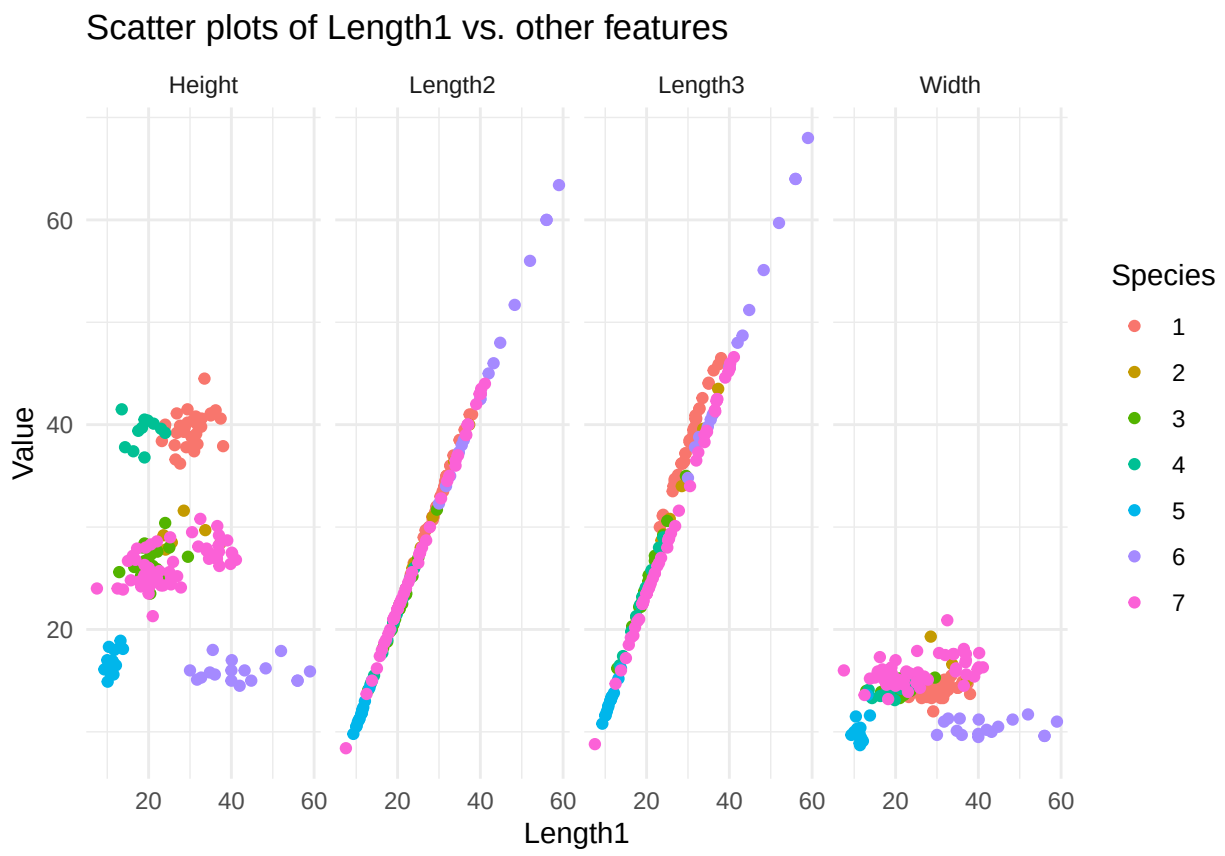
Species	mean L1	mean L2	mean L3	mean H	mean W	SD L1	SD L2	SD L3	SD H	SD W
1	30.33	33.14	38.39	39.59	14.15	3.64	3.97	4.22	1.60	0.78
2	28.80	31.32	34.32	29.20	15.90	5.58	5.72	6.02	1.35	1.82
3	20.64	22.27	24.97	26.74	14.61	3.46	3.65	4.03	1.49	0.78
4	18.73	20.35	22.79	39.31	14.08	3.28	3.56	3.96	1.43	0.60
5	11.26	11.92	13.04	16.89	10.22	1.22	1.43	1.43	1.13	1.29
6	42.48	45.48	48.72	15.84	10.44	9.03	9.71	10.17	1.00	0.75
7	25.74	27.89	29.57	26.26	15.84	8.56	9.02	9.53	1.91	1.36

Visualisierung

Im folgenden sind Diagramme dargestellt, welche die Abhängigkeiten der einzelnen Variablen untereinander darstellen.

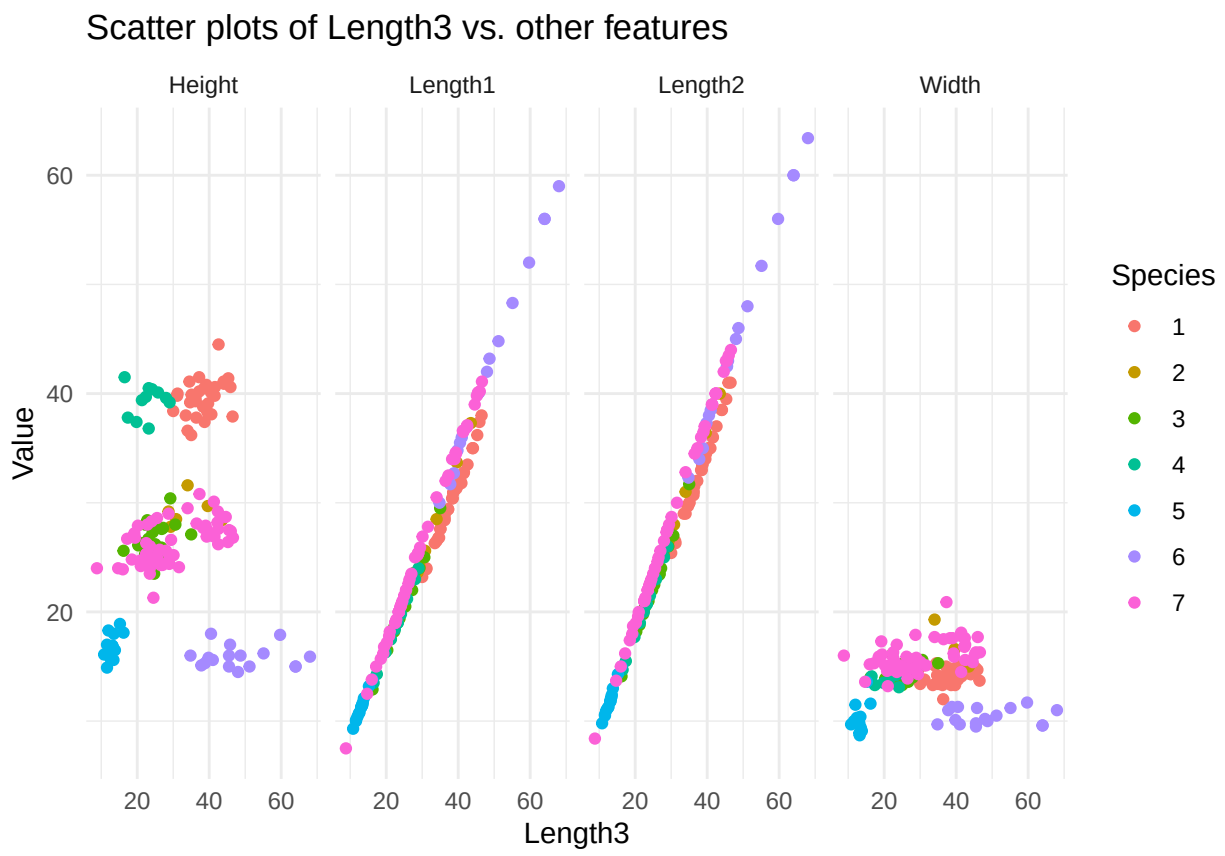
```r
# Create a long-format data frame
long_fish_1_df <- fish_df %>%
  dplyr::select(Species, Length1, Length2, Length3, Height, Width) %>%
  tidyr::gather(key = "Feature", value = "Value", -c(Species, Length1))

long_fish_2_df <- fish_df %>%
  dplyr::select(Species, Length1, Length2, Length3, Height, Width) %>%
  tidyr::gather(key = "Feature", value = "Value", -c(Species, Length2))

long_fish_3_df <- fish_df %>%
  dplyr::select(Species, Length1, Length2, Length3, Height, Width) %>%
  tidyr::gather(key = "Feature", value = "Value", -c(Species, Length3))

long_fish_4_df <- fish_df %>%
  dplyr::select(Species, Length1, Length2, Length3, Height, Width) %>%
  tidyr::gather(key = "Feature", value = "Value", -c(Species, Height))

long_fish_5_df <- fish_df %>%
  dplyr::select(Species, Length1, Length2, Length3, Height, Width) %>%
  tidyr::gather(key = "Feature", value = "Value", -c(Species, Width))

ggplot(long_fish_1_df, aes(x = Length1, y = Value, color = factor(Species))) +
  geom_point() +
  labs(x = "Length1", y = "Value", color = "Species") +
```

```
  theme_minimal() +
  facet_wrap(~Feature, ncol = 4) +
  ggtitle("Scatter plots of Length1 vs. other features")
```

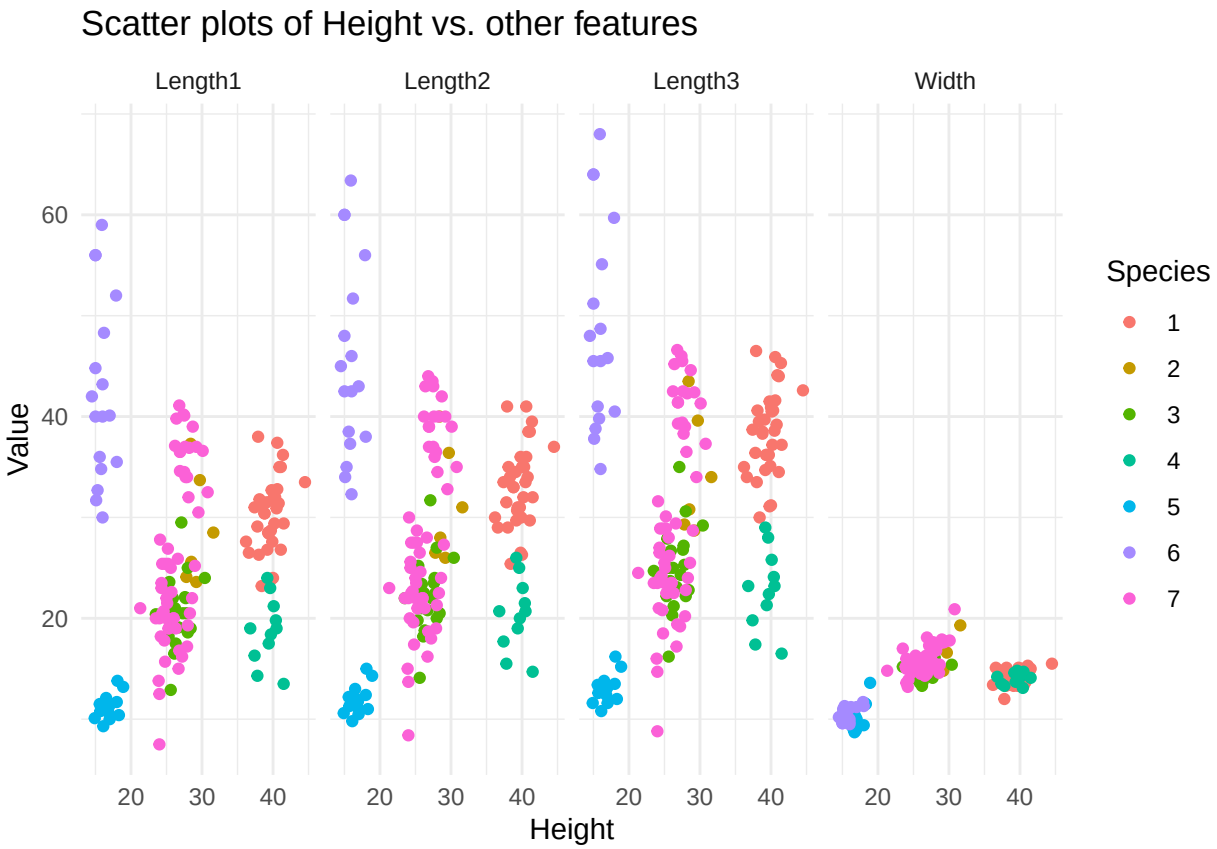



```
ggplot(long_fish_2_df, aes(x = Length2, y = Value, color = factor(Species))) +
  geom_point() +
  labs(x = "Length2", y = "Value", color = "Species") +
  theme_minimal() +
  facet_wrap(~Feature, ncol = 4) +
  ggtitle("Scatter plots of Length2 vs. other features")
```

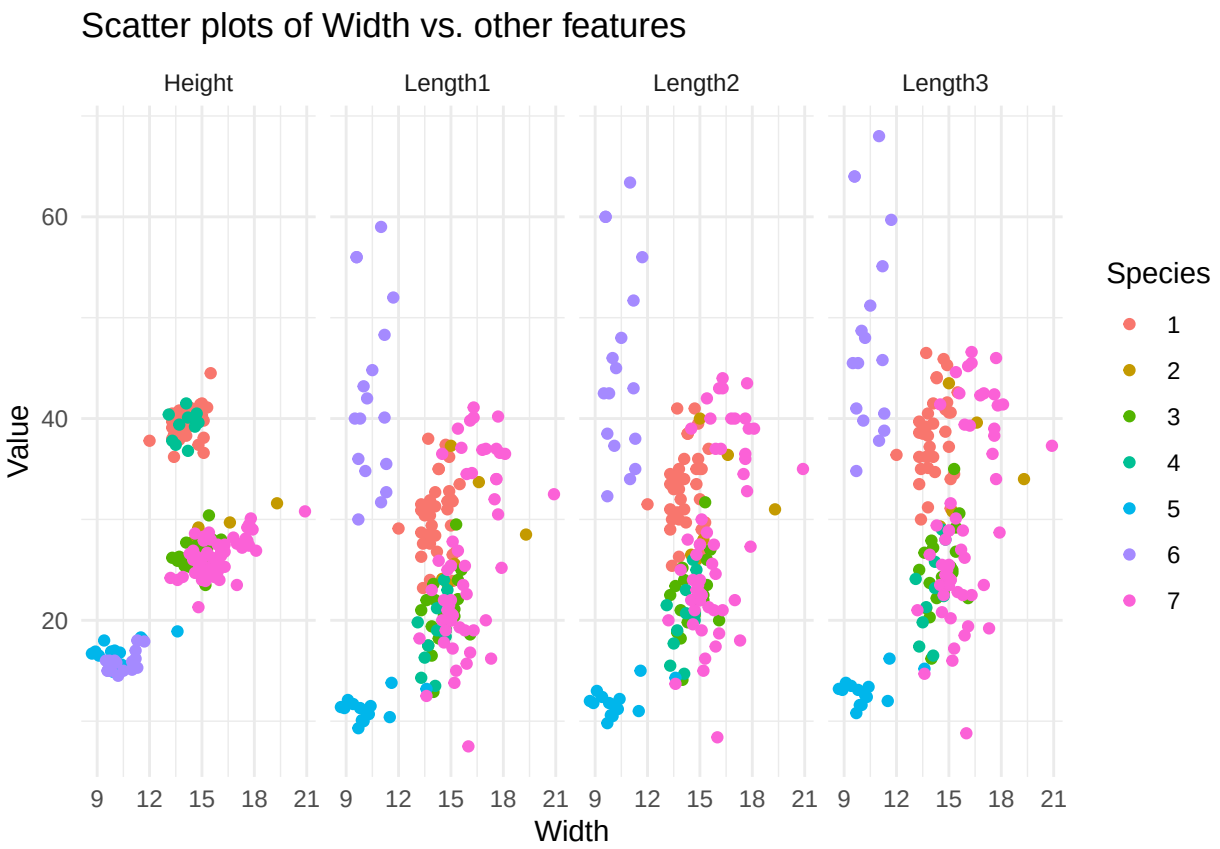

```
ggplot(long_fish_3_df, aes(x = Length3, y = Value, color = factor(Species))) +
  geom_point() +
  labs(x = "Length3", y = "Value", color = "Species") +
  theme_minimal() +
  facet_wrap(~Feature, ncol = 4) +
  ggtitle("Scatter plots of Length3 vs. other features")
```




```
ggplot(long_fish_4_df, aes(x = Height, y = Value, color = factor(Species))) +
  geom_point() +
  labs(x = "Height", y = "Value", color = "Species") +
  theme_minimal() +
  facet_wrap(~Feature, ncol = 4) +
  ggtitle("Scatter plots of Height vs. other features")
```




```
ggplot(long_fish_5_df, aes(x = Width, y = Value, color = factor(Species))) +
  geom_point() +
  labs(x = "Width", y = "Value", color = "Species") +
  theme_minimal() +
  facet_wrap(~Feature, ncol = 4) +
  ggtitle("Scatter plots of Width vs. other features")
```



Klassifikator

Zunächst werden die einzigartigen Fischarten aus den Trainingsdaten ermittelt. Für jede Fischart werden daraufhin die Mittelwerte, Kovarianzmatrizen und a-priori Wahrscheinlichkeiten berechnet.

Anschließend wird die Dichte der multivariaten Gaußverteilung für jeden gegebenen Datenpunkt und die zugehörigen Verteilungsparameter ermittelt.

Die LDA- und QDA-Classifier nutzen die Maximum-Likelihood-Methode, um Fische basierend auf den Merkmalen zu klassifizieren. Hierbei wird für jeden Datenpunkt die bedingte Wahrscheinlichkeit jeder Fischart berechnet und mit der entsprechenden a-priori Wahrscheinlichkeit multipliziert. Die Fischart mit der höchsten resultierenden Wahrscheinlichkeit wird als vorhergesagte Spezies ausgewählt. Der Unterschied zwischen der LDA- und QDA-Methode liegt in der Annahme bezüglich der Kovarianzmatrizen: Bei der LDA wird eine gemeinsame Kovarianzmatrix für alle Klassen angenommen, während bei der QDA jeder Klasse eine individuelle Kovarianzmatrix zugeordnet wird. Dies führt dazu, dass QDA-Classifier flexibler sind und komplexere Trennlinien zwischen den Klassen ermöglichen, während LDA-Classifier weniger anfällig für Überanpassung sind, sofern die Annahme einer gemeinsamen Kovarianzmatrix zutreffend ist.

Abschließend wird der Klassifikator auf den Testdatensatz angewendet, und die Genauigkeit ermittelt, indem die Anzahl der korrekt vorhergesagten Spezies durch die Gesamtanzahl der Vorhersagen geteilt wird.

```
# Calculate the mean, covariance, and a-priori probability for each fish species
species_list <- unique(train_data$Species)
stats_list <- lapply(species_list, function(species) {
  species_data <- train_data[train_data$Species ==
                             species, c("Length1", "Length2",
```

```
                                          "Length3", "Height", "Width")]
  n <- nrow(species_data)

  list(mean = colMeans(species_data),
       cov = cov(species_data) + diag(1e-6, ncol(species_data)),
       prior = n / nrow(train_data))
})

names(stats_list) <- species_list

# Multivariate Gaussian density function
multi_gau <- function(x, mean, cov) {
  k <- length(mean)
  exp(-0.5 * t(x - mean) %*% solve(cov, x - mean)) / sqrt((2 * pi)^k * det(cov))
}

### QDA ###

# classifier
classifier_fish_qda <- function(lengths) {
  likelihoods <- sapply(stats_list, function(params) {
    likelihood <- multi_gau(lengths, params$mean, params$cov)
    likelihood * params$prior
  })

  names(likelihoods)[which.max(likelihoods)]
}

# Classify each fish in the test set
predicted_species_qda <- apply(test_data[, c("Length1", "Length2", "Length3",
                                             "Height", "Width")],
                               1, classifier_fish_qda)

### LDA ###

# Calculate the common covariance matrix for LDA
common_cov <- cov(train_data[, c("Length1", "Length2", "Length3", "Height",
                                 "Width")]) +
  diag(1e-6, ncol(train_data[, c("Length1",
                                 "Length2", "Length3", "Height", "Width")]))

# LDA classifier
classifier_fish_lda <- function(lengths) {
  likelihoods <- sapply(stats_list, function(params) {
    likelihood <- multi_gau(lengths, params$mean, common_cov)
    likelihood * params$prior
  })

  names(likelihoods)[which.max(likelihoods)]
}

# Classify each fish in the test set
predicted_species_lda <- apply(test_data[, c("Length1",
```

```
                                          "Length2", "Length3",
                                          "Height", "Width")],
                                  1, classifier_fish_lda)


# summary of species in test data and predicted species
comparision_qda_df <- data.frame(Species = test_data$Species,
                                 Prediction = predicted_species_qda)
comparision_lda_df <- data.frame(Species = test_data$Species,
                                 Prediction = predicted_species_lda)

# Calculate accuracy
correct_predictions_lda <- sum(predicted_species_lda == test_data$Species)
correct_predictions_qda<- sum(predicted_species_qda == test_data$Species)

# total predictions
total_predictions <- length(predicted_species_lda)


accuracy_lda <- correct_predictions_lda / total_predictions
accuracy_qda <- correct_predictions_qda / total_predictions
```

In der folgenden Tabelle, ist die Maximum-Liklyhood zusammengefasst. Grüne Zellen wurde korrekt vorhergesagt, rosane falsch.

```
# LDA classifier with likelihood percentages
classifier_fish_lda_percent <- function(lengths) {
  likelihoods <- sapply(stats_list, function(params) {
    likelihood <- multi_gau(lengths, params$mean, common_cov)
    likelihood * params$prior
  })

  likelihood_percentages <- likelihoods / sum(likelihoods) * 100
  return(likelihood_percentages)
}

# QDA classifier with likelihood percentages
classifier_fish_qda_percent <- function(lengths) {
  likelihoods <- sapply(stats_list, function(params) {
    likelihood <- multi_gau(lengths, params$mean, params$cov)
    likelihood * params$prior
  })

  likelihood_percentages <- likelihoods / sum(likelihoods) * 100
  return(likelihood_percentages)
}

# Apply the modified classifiers to the test data
lda_likelihood_percentages <- t(apply(test_data[, c("Length1",
                                                    "Length2", "Length3",
                                                    "Height", "Width")],
                                      1, classifier_fish_lda_percent))
qda_likelihood_percentages <- t(apply(test_data[, c("Length1",
```

```
                                                    "Length2", "Length3",
                                                    "Height", "Width")],
                                        1, classifier_fish_qda_percent))

# Create data frames for the likelihood percentages
lda_percent_df <- data.frame(lda_likelihood_percentages,
                             Species = test_data$Species)
qda_percent_df <- data.frame(qda_likelihood_percentages,
                             Species = test_data$Species)

# Number of decimal places you want to round to
decimal_places <- 1000

# Round all numbers in the data frame, excluding the Species column
lda_percent_df_rounded <-
  as.data.frame(lapply(lda_percent_df[, -ncol(lda_percent_df)],
                       function(x) round(x, decimal_places)))
lda_percent_df_rounded$Species <- lda_percent_df$Species

# Round all numbers in the data frame, excluding the Species column
qda_percent_df_rounded <-
  as.data.frame(lapply(qda_percent_df[, -ncol(qda_percent_df)],
                       function(x) round(x, decimal_places)))
qda_percent_df_rounded$Species <- qda_percent_df$Species

# Format all numbers in the data frame, excluding the Species column
lda_percent_df_sci <-
  as.data.frame(lapply(lda_percent_df_rounded[, -ncol(lda_percent_df_rounded)],
                       function(x) formatC(x, format = "e", digits = 2)))
lda_percent_df_sci$Species <- lda_percent_df_rounded$Species

# Format all numbers in the data frame, excluding the Species column
qda_percent_df_sci <-
  as.data.frame(lapply(qda_percent_df_rounded[, -ncol(qda_percent_df_rounded)],
                       function(x) formatC(x, format = "e", digits = 2)))
qda_percent_df_sci$Species <- qda_percent_df_rounded$Species

lda_percent_df_sci$Prediction <- predicted_species_lda
qda_percent_df_sci$Prediction <- predicted_species_qda

# find the column with the max value for each row
lda_percent_df_sci$max_col <- apply(lda_percent_df_sci, 1,
                                    function(x) which.max(as.numeric(x)))

# create gt table
lda_percent_gt_sci <- gt(lda_percent_df_sci)

# iterate over rows of the table to color the max value cell
for (i in seq_len(nrow(lda_percent_df_sci))) {
  lda_percent_gt_sci <- tab_style(
    lda_percent_gt_sci,
    style = cell_fill(color = "lightgreen"),
    locations = cells_body(
```

```
      columns = lda_percent_df_sci$max_col[i],
      rows = i
    )
  )
}

# changes to table
lda_percent_gt_sci <-
  lda_percent_gt_sci %>%
  cols_hide(columns = "max_col") %>%
  # Title
  tab_header(
    title = "Maximum-Liklyhood-Selections LDA-Classifier",
    subtitle = "in Percent [%]"
  ) %>%
  # color cell
  tab_style(
    style = cell_fill(color = "lightpink"),
    locations = cells_body(
      columns = c(7),
      rows = 6)
    ) %>%
  tab_style(
    style = cell_fill(color = "lightgoldenrodyellow"),
    locations = cells_body(
      columns = c(3),
      rows = 6)
    )

qda_percent_df <- qda_percent_df_sci

# find the column with the max value for each row
qda_percent_df$max_col <- apply(qda_percent_df, 1,
                                function(x) which.max(as.numeric(x)))

# create gt table
qda_percent_gt_sci <- gt(qda_percent_df)

# iterate over rows of the table to color the max value cell
for (i in seq_len(nrow(qda_percent_df))) {
  qda_percent_gt_sci <- tab_style(
    qda_percent_gt_sci,
    style = cell_fill(color = "lightgreen"),
    locations = cells_body(
      columns = qda_percent_df$max_col[i],
      rows = i
    )
  )
}
# changes to table
qda_percent_gt_sci <-
  qda_percent_gt_sci %>%
  cols_hide(columns = "max_col") %>%
```

```
  # Title
  tab_header(
    title = "Maximum-Liklyhood-Selections QDA-Classifier",
    subtitle = "in Percent [%]"
  )

lda_percent_gt_sci
```

Maximum-Liklyhood-Selections LDA-Classifier

in Percent [%]

X1	X2	X3	X4	X5	X6	X7	Species	Prediction
8.36e+01	1.41e+00	8.10e+00	7.42e-01	8.09e-01	1.44e-01	5.15e+00	1	1
8.74e+01	1.54e+00	6.86e+00	1.68e-01	1.86e-01	1.47e-01	3.70e+00	1	1
8.77e+01	6.49e-01	4.73e+00	2.17e+00	2.13e-01	2.75e-01	4.30e+00	1	1
9.24e+01	2.19e+00	5.05e-01	1.02e-01	1.63e-02	3.20e-01	4.48e+00	1	1
6.15e+00	4.35e+00	5.58e+01	5.86e-02	1.06e+01	1.10e-01	2.29e+01	3	3
1.34e+01	1.34e+01	3.14e+01	1.94e-02	1.40e+00	2.35e-01	4.02e+01	3	7
1.06e-01	2.72e-03	1.35e-02	9.93e+01	2.58e-02	1.87e-04	5.12e-01	4	4
9.98e+00	3.95e-01	1.31e+00	6.90e+01	2.88e-01	5.58e-02	1.90e+01	4	4
5.76e-01	2.87e-01	1.27e+01	2.94e-02	8.18e+01	6.69e-01	3.93e+00	5	5
1.01e+00	2.22e-01	5.93e-01	4.49e-02	2.89e+00	8.91e+01	6.14e+00	6	6
6.07e-02	9.20e-03	3.84e-03	1.69e-04	4.58e-03	9.97e+01	1.75e-01	6	6
2.13e+00	5.67e+00	3.67e+00	2.25e-01	1.63e+00	2.53e-01	8.64e+01	7	7
1.07e+00	1.60e+01	1.19e+01	2.14e-03	2.35e-01	5.57e-02	7.07e+01	7	7
1.21e+00	5.64e+00	3.71e+00	6.49e-02	3.77e-01	2.53e-01	8.87e+01	7	7
9.62e-01	3.63e+00	3.66e+00	9.59e-02	3.78e-02	1.46e-01	9.15e+01	7	7
1.68e+00	5.28e+00	1.95e+00	4.20e-02	2.17e-02	6.20e-01	9.04e+01	7	7

```
qda_percent_gt_sci
```

Maximum-Liklyhood-Selections QDA-Classifier

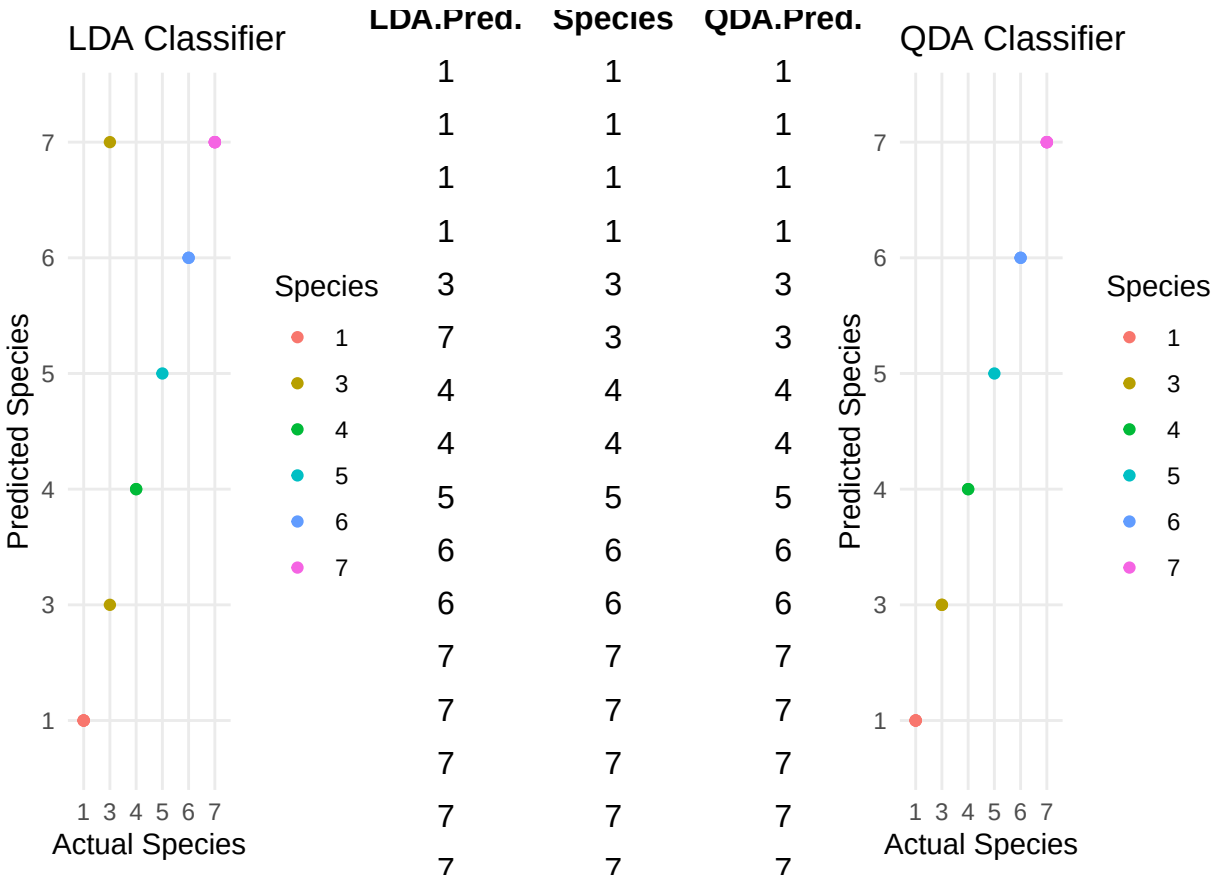
in Percent [%]

X1	X2	X3	X4	X5	X6	X7	Species	Prediction
1.00e+02	0.00e+00	8.38e-21	0.00e+00	0.00e+00	3.88e-290	1.00e-81	1	1
1.00e+02	0.00e+00	9.49e-22	0.00e+00	0.00e+00	5.93e-313	2.32e-94	1	1
1.00e+02	0.00e+00	7.02e-25	0.00e+00	0.00e+00	0.00e+00	1.36e-91	1	1
1.00e+02	0.00e+00	5.96e-26	1.24e-174	0.00e+00	0.00e+00	2.09e-92	1	1
3.43e-30	0.00e+00	1.00e+02	6.27e-61	1.42e-51	1.43e-37	5.70e-09	3	3
1.44e-25	0.00e+00	1.00e+02	3.24e-22	4.46e-69	1.37e-54	1.26e-12	3	3
2.12e-29	0.00e+00	2.89e-35	1.00e+02	5.57e-175	5.48e-308	7.65e-38	4	4
2.53e-14	0.00e+00	1.33e-17	1.00e+02	4.99e-142	5.43e-255	1.06e-32	4	4
9.70e-68	0.00e+00	2.44e-11	1.07e-63	1.00e+02	3.82e-03	3.19e-12	5	5
6.19e-80	0.00e+00	1.57e-48	0.00e+00	3.90e-269	1.00e+02	6.91e-14	6	6
7.50e-102	0.00e+00	8.55e-121	0.00e+00	0.00e+00	1.00e+02	1.53e-25	6	6
2.63e-58	0.00e+00	4.42e-04	0.00e+00	1.59e-08	6.90e-28	1.00e+02	7	7
1.90e-70	0.00e+00	3.05e-07	8.66e-296	1.48e-10	8.91e-32	1.00e+02	7	7
5.74e-69	0.00e+00	2.27e-10	0.00e+00	2.52e-34	2.04e-28	1.00e+02	7	7
3.62e-72	0.00e+00	1.24e-19	0.00e+00	2.10e-151	5.61e-57	1.00e+02	7	7

1.77e-65	0.00e+00	7.02e-27	0.00e+00	5.13e-203	4.20e-48	1.00e+02	7	7

Wie in der folgenden Graphik zu sehen ist, ist die Genauigkeit des QDA-Classifiers höher (100%), als die des LDA-Classifiers, welcher eine Genauigkeit von 93,75% aufweist.

```
# For LDA classifier
comparision_df_lda <- data.frame(Species = test_data$Species,
                                 Prediction = predicted_species_lda)

# For QDA classifier
comparision_df_qda <- data.frame(Species = test_data$Species,
                                 Prediction = predicted_species_qda)

# Create scatter plots for LDA and QDA
plot_lda <- ggplot(comparision_df_lda, aes(x = Species, y =
                                           Prediction, color = Species)) +
  geom_point() +
  labs(title = "LDA Classifier", x = "Actual Species", y =
         "Predicted Species") +
  theme_minimal()

plot_qda <- ggplot(comparision_df_qda, aes(x = Species, y =
                                           Prediction, color = Species)) +
  geom_point() +
  labs(title = "QDA Classifier", x = "Actual Species", y = "Predicted Species") +
  theme_minimal()

data_table <- data.frame("LDA Pred." = predicted_species_lda,
                         Species = test_data$Species,
                         "QDA Pred." = predicted_species_qda)

table_theme <- ttheme_minimal(
  core = list(fg_params = list(hjust = 0, x = 0.1)),
  colhead = list(fg_params = list(hjust = 0, x = 0.1))
)

# Convert the data frame to a tableGrob
table_grob <- tableGrob(data_table,  theme = tt2, rows = NULL)
# Adjust the table size

# Display plots side by side with the table of Species and Predictions
grid.arrange(plot_lda, table_grob, plot_qda, ncol = 3)
```



LDA.Pred.	Species	QDA.Pred.
1	1	1
1	1	1
1	1	1
1	1	1
3	3	3
7	3	3
4	4	4
4	4	4
5	5	5
6	6	6
6	6	6
7	7	7
7	7	7
7	7	7
7	7	7
7	7	7

Quellen

[1] Vorlesungsscript Datenerfassung in Umweltinformationssystemen

[2] Trevor Hastie Robert Tibshirani Jerome Friedman. The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer Series in Statistics. 2009

[3] Scikit-learn. Linear and Quadratic Discriminant Analysis. Zugriffszeit: 08.05.2023 17:45 Uhr. https://scikit-learn.org/stable/modules/lda_qda.html

Packages

Es wurden folgende Packages verwendet:

tidyverse
ggplot2
zoo
scales
lubridate
mvtnorm
caret
grid
gridExtra
gt