Project

Jordan Klein

2025-12-02

Helper Functions

This functions will be used for preprocessing and analysis.

```r
impute_lm <- function(feature, data, condition) {

  # Impute values in a numeric column using a linear model.
  # Rows where `condition` is FALSE are used to train the model.
  # For rows where `condition` is TRUE, the function predicts and replaces the value.
  # All other rows keep their original values.

  feature   <- rlang::enquo(feature)
  condition <- rlang::enquo(condition)

  # Rows where condition is FALSE
  train_data <- data %>%
    dplyr::filter(!(!!condition))

  # Fit LM
  formula <- as.formula(paste0(rlang::as_name(feature), " ~ ."))
  fit <- lm(formula, data = train_data)

  # Predict for rows needing imputation
  pred_data <- data %>% dplyr::filter(!!condition)
  preds <- predict(fit, newdata = pred_data)

  # Original column
  out <- data[[rlang::as_name(feature)]]

  # Correctly evaluate logical condition within `data`
  cond_logical <- rlang::eval_tidy(condition, data)

  # Replace
  out[cond_logical] <- preds

  out
}

get_predictions <- function(
    formula,
    train,
    test,
    model = c("lm", "rf"),
    ...
) {

  # Train a linear model or random forest on `train` and generate predictions for `test`.
  # Select model via `model = "lm"` or `"rf"`.
  # Returns a vector of predictions.


  # match the argument + enforce allowed values
  model <- match.arg(model)


  if (model == "lm") {
    # Linear Model
    fit <- lm(formula, data = train, ...)
    preds <- predict(fit, newdata = test)

  } else if (model == "rf") {
    # Random Forest
    fit <- ranger::ranger(
      formula,
      data = train,
      num.threads = parallel::detectCores(),
      ...
    )
    preds <- predict(fit, test)$predictions
  }

  return(preds)
}

kfold_metrics <- function(
    formula,
    data,
    k = 10,
    model = c("lm", "rf"),
    ...
) {

  # Perform k-fold cross-validation for a given formula and model type (lm or rf).
  # Trains on k-1 folds and predicts the held-out fold, repeating for all k folds.
  # Returns MAE and MAPE along with the model type and formula used.

  model <- match.arg(model)

  # Create folds
  folds <- createFolds(1:nrow(data), k = k, list = TRUE)

  preds  <- numeric(nrow(data))
  actual <- model.response(model.frame(formula, data))

  # Perform CV
  for (i in seq_along(folds)) {
    test_idx  <- folds[[i]]
    train_idx <- setdiff(seq_len(nrow(data)), test_idx)

    train <- data[train_idx, ]
    test  <- data[test_idx, ]

    # use your helper
    preds[test_idx] <- get_predictions(
      formula = formula,
      train   = train,
      test    = test,
      model   = model,
      ...
    )
  }

  # Metrics
  mae  <- mean(abs(preds - actual))
  mape <- mean(abs((actual - preds) / actual)) * 100

  data.frame(
    model   = model,
    formula = paste(deparse(formula), collapse = ""),
    mae     = mae,
    mape    = mape
  )
}


eval_circuit <- function(
    formulas,
    data,
    k = 10,
    excel_path = NULL,
    ...
) {

  # -----------------------------------------------------------
  # Eval Circuit
  # Runs model evaluation across all provided formulas.
  #
  # Given:
  #   - formulas: a named list of formulas
  #   - data: dataset
  #   - k: number of folds for k-fold CV
  #   - excel_path: optional path to save results as an Excel file
  #
  # Process:
  #   • For each formula:
  #       - Run k-fold CV using Linear Regression
  #       - Run k-fold CV using Random Forest
  #       - Store MAE, MAPE, model type, and the formula name
  #   • Combine results for all formulas and models
  #   • Sort final table by MAE (ascending)
  #   • Optionally export the results to Excel
  #
  # Returns:
  #   A data frame with:
  #       formula_name, model type, mae, mape, and other metrics
  # -----------------------------------------------------------

  results <- Map(function(name, form) {

    lm_res <- kfold_metrics(
      formula = form,
      data    = data,
      k       = k,
      model   = "lm",
      ...
    )
    lm_res$formula_name <- name     # <--- save the name

    rf_res <- kfold_metrics(
      formula = form,
      data    = data,
      k       = k,
      model   = "rf",
      ...
    )
    rf_res$formula_name <- name     # <--- same

    rbind(lm_res, rf_res)

  }, names(formulas), formulas)

  # Combine all
  results_df <- do.call(rbind, results)

  # Order by MAE
  results_df <- results_df[order(results_df$mae), ]
  rownames(results_df) <- NULL

  # Save Excel if needed
  if (!is.null(excel_path)) {
    write_xlsx(results_df, excel_path)
    message("Results written to: ", excel_path)
  }

  return(results_df)
}
```

Load and Preprocess

Impute missing values, engineer features of EDA and modeling

```r
data <- read.csv("train2.csv")

Features <- list(
  Target = "Age",
  Size = c("Length", "Height", "Diameter"),
  Mass = c("Weight", "Shucked.Weight", "Viscera.Weight", "Shell.Weight")
)
Features$All <- c(Features$Size, Features$Mass)

abalone <- data %>%
  mutate(
    Volume = 4/3 * pi * Length/2 * Height/2 * Diameter/2,
    IsAdult = factor(if_else(Sex == "I", "Infant", "Adult"),
                     ordered = TRUE, levels = c("Infant", "Adult")),
    Sex = factor(case_when(
      Sex == "I" ~ "Infant",
      Sex == "F" ~ "Female",
      Sex == "M" ~ "Male"
    )),
    Height = impute_lm(
      feature = Height,
      data = cur_data(),
      condition = Height == 0
    ),
    Diameter = impute_lm(
      feature = Diameter,
      data = cur_data(),
      condition = Diameter == 0
    ),
    Shucked.Weight = impute_lm(
      feature = Shucked.Weight,
      data = cur_data(),
      condition = Shucked.Weight > Weight
    ),
    Weight.Mean1 = (Shucked.Weight*Viscera.Weight*Shell.Weight)**(1/3),
    Weight.Mean2 = (Weight*Shucked.Weight*Viscera.Weight*Shell.Weight)**(1/4),
    Weight.Norm1 = sqrt(Shucked.Weight**2 + Viscera.Weight**2 + Shell.Weight**2),
    Weight.Norm2 = sqrt(Shucked.Weight**2 + Viscera.Weight**2 + Shell.Weight**2 + Weight **2),
    Size.Norm = sqrt(Length**2 + Diameter**2 + Height**2),
    Size.Mean = (Length + Diameter + Height)**(1/3),
    Shell.Ratio = Shell.Weight / Weight,
    Meat.Ratio = Shucked.Weight / Weight,
    Viscera.Ratio = Viscera.Weight / Weight,
    Soft.Ratio = (Shucked.Weight + Viscera.Weight) /Weight,
    Elongation = Length / Diameter,
    Flatness = Height / sqrt(Length*Diameter),
    SurfaceArea = (Length * Diameter + Length * Height + Diameter * Height),
    Roundness = Diameter / Length,
    Density = Volume / Weight
  )
```

```
## Warning: There was 1 warning in ‘mutate()‘.
## i In argument: ‘Height = impute_lm(...)‘.
## Caused by warning:
## ! ‘cur_data()‘ was deprecated in dplyr 1.1.0.
## i Please use ‘pick()‘ instead.
```

```r
summary(abalone)
```

```
##        id             Sex          Diameter          Length     
##  Min.   :    1   Female:4576   Min.   :0.2000   Min.   :0.200  
##  1st Qu.: 3751   Infant:4967   1st Qu.:0.8875   1st Qu.:1.150  
##  Median : 7500   Male  :5457   Median :1.0750   Median :1.375  
##  Mean   : 7500                 Mean   :1.0221   Mean   :1.315  
##  3rd Qu.:11250                 3rd Qu.:1.2000   3rd Qu.:1.538  
##  Max.   :15000                 Max.   :1.5750   Max.   :2.038  
##      Height           Weight        Shucked.Weight    Viscera.Weight    
##  Min.   :0.0250   Min.   : 0.2268   Min.   : 0.0245   Min.   : 0.01417  
##  1st Qu.:0.2875   1st Qu.:13.1967   1st Qu.: 5.6557   1st Qu.: 2.80660  
##  Median :0.3625   Median :23.5584   Median : 9.8585   Median : 4.89029  
##  Mean   :0.3468   Mean   :23.2434   Mean   :10.0330   Mean   : 5.01447  
##  3rd Qu.:0.4125   3rd Qu.:32.1625   3rd Qu.:13.9763   3rd Qu.: 6.98815  
##  Max.   :0.6000   Max.   :75.3246   Max.   :42.1841   Max.   :20.12814  
##   Shell.Weight          Age             Volume          IsAdult     
##  Min.   : 0.09922   Min.   : 1.000   Min.   :0.0000   Infant: 4967  
##  1st Qu.: 3.82718   1st Qu.: 8.000   1st Qu.:0.1548   Adult :10033  
##  Median : 6.80388   Median :10.000   Median :0.2829                 
##  Mean   : 6.67329   Mean   : 9.985   Mean   :0.2813                 
##  3rd Qu.: 9.07184   3rd Qu.:11.000   3rd Qu.:0.3953                 
##  Max.   :29.10076   Max.   :29.000   Max.   :0.8821                 
##   Weight.Mean1       Weight.Mean2       Weight.Norm1      Weight.Norm2    
##  Min.   : 0.06579   Min.   : 0.09683   Min.   : 0.1671   Min.   : 0.3414  
##  1st Qu.: 3.96474   1st Qu.: 5.35190   1st Qu.: 7.5146   1st Qu.:15.2543  
##  Median : 6.97497   Median : 9.47937   Median :13.1642   Median :27.0724  
##  Mean   : 6.91257   Mean   : 9.35784   Mean   :13.1161   Mean   :26.6984  
##  3rd Qu.: 9.61440   3rd Qu.:13.01581   3rd Qu.:18.2722   3rd Qu.:37.0408  
##  Max.   :22.89460   Max.   :30.53401   Max.   :48.2741   Max.   :89.4661  
##    Size.Norm        Size.Mean       Shell.Ratio       Meat.Ratio     
##  Min.   :0.3562   Min.   :0.8298   Min.   :0.1383   Min.   :0.08231  
##  1st Qu.:1.4808   1st Qu.:1.3248   1st Qu.:0.2679   1st Qu.:0.39884  
##  Median :1.7897   Median :1.4136   Median :0.2887   Median :0.43269  
##  Mean   :1.7014   Mean   :1.3800   Mean   :0.2924   Mean   :0.43054  
##  3rd Qu.:1.9899   3rd Qu.:1.4659   3rd Qu.:0.3137   3rd Qu.:0.46344  
##  Max.   :2.6282   Max.   :1.6054   Max.   :1.0000   Max.   :0.82876  
##  Viscera.Ratio       Soft.Ratio       Elongation        Flatness     
##  Min.   :0.01587   Min.   :0.1746   Min.   :0.6667   Min.   :0.0603  
##  1st Qu.:0.19867   1st Qu.:0.6116   1st Qu.:1.2632   1st Qu.:0.2790  
##  Median :0.21463   Median :0.6494   Median :1.2885   Median :0.2964  
##  Mean   :0.21600   Mean   :0.6465   Mean   :1.2925   Mean   :0.2975  
##  3rd Qu.:0.23270   3rd Qu.:0.6841   3rd Qu.:1.3165   3rd Qu.:0.3146  
##  Max.   :0.76190   Max.   :1.2381   Max.   :2.2500   Max.   :0.5653  
##   SurfaceArea        Roundness         Density       
##  Min.   :0.09977   Min.   :0.4444   Min.   :0.00000  
##  1st Qu.:1.60988   1st Qu.:0.7596   1st Qu.:0.01110  
##  Median :2.38016   Median :0.7761   Median :0.01196  
##  Mean   :2.26747   Mean   :0.7748   Mean   :0.01196  
##  3rd Qu.:2.96352   3rd Qu.:0.7917   3rd Qu.:0.01284  
##  Max.   :5.10562   Max.   :1.5000   Max.   :0.02557
```

EVALUATE DATA QUALITY

Some features have missing data. Diameter and Height have 0 values. Some rows have Shucked Weight which are higher than the total weight.

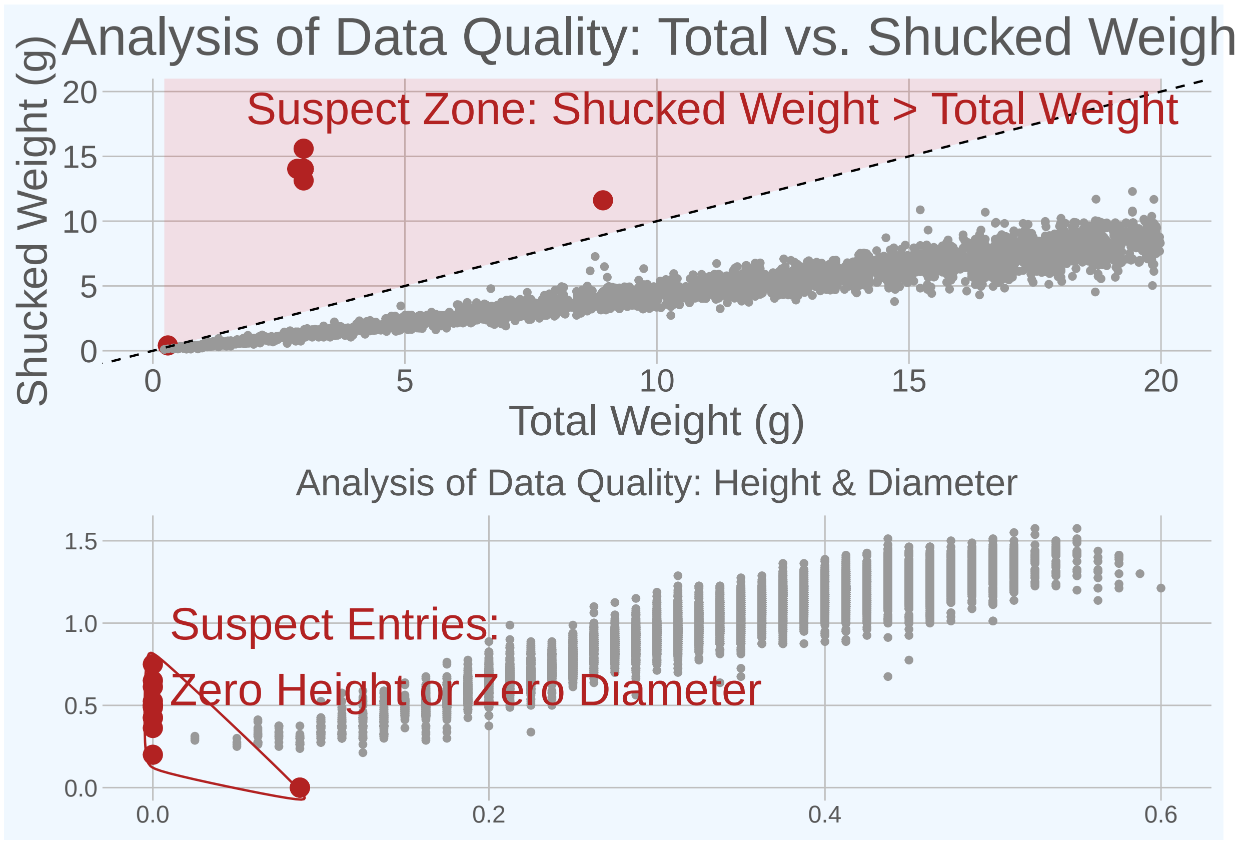
This values are impossible, and will be replaced using a predictive model that considers all other features.

```
data <- read.csv("train2.csv")

p1<- data %>%
  select(Weight, Shucked.Weight) %>%
  mutate(
    Suspect = Shucked.Weight > Weight
  ) %>%
  ggplot(aes(x = Weight, y = Shucked.Weight)) +
    # Shade area below y = x
    geom_ribbon(aes(ymin = Weight, ymax = Inf), fill = "red", alpha = 0.1) +
    # Points
    geom_point(aes(color = Suspect, size = Suspect)) +
    scale_color_manual(values = c("FALSE" = "gray60", "TRUE" = "firebrick")) +
    scale_size_manual(values = c("FALSE" = 1, "TRUE" = 3)) +
    # Optional: draw y = x line for reference
    geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
    annotate(
      "text",
      x = 0,
      y = 20,
      label = "Suspect Zone: Shucked Weight > Total Weight",
      hjust = -.1,          # left-align text at x = 0
      vjust = 1,          # anchor at top of plot
      size = 6,
      color = "firebrick"
    ) +
    scale_y_continuous(limits = c(0,20)) +
    scale_x_continuous(limits = c(0,20)) +
    theme_excel_new() +
    labs(title = "Analysis of Data Quality: Total vs. Shucked Weight",
         x = "Total Weight (g)",
         y = "Shucked Weight (g)") +
    theme(
      plot.title = element_text(size = 20),
      axis.title = element_text(size = 16),
      axis.text = element_text(size = 12),
      legend.position = "none",
      plot.background  = element_rect(fill = "aliceblue", color = NA),
      panel.background = element_rect(fill = "aliceblue", color = NA)
    )
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## i The deprecated feature was likely used in the ggthemes package.
##   Please report the issue at <https://github.com/jrnold/ggthemes/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
p2 <- data %>%
  select(Diameter, Height) %>%
  mutate(
    Suspect = Diameter == 0 | Height == 0
  ) %>%
  ggplot(aes(x = Height, y = Diameter)) +

  # Points
  geom_point(aes(color = Suspect, size = Suspect)) +
  scale_color_manual(values = c("FALSE" = "gray60", "TRUE" = "firebrick")) +
  scale_size_manual(values = c("FALSE" = 1, "TRUE" = 3)) +

  geom_encircle(
    data = function(df) dplyr::filter(df, Suspect),
    aes(x = Height, y = Diameter),
    color = "firebrick",
    size = 1.2,
    expand = 0.04,
    s_shape = 0.8
  ) +

  # Labels
  labs(
    title = "Analysis of Data Quality: Height & Diameter",
    x = "Height (mm)",
    y = "Diameter (mm)"
  ) +
  annotate(
    "text",
    x = 0.01, y = 0.8,
    label = "Suspect Entries:\nZero Height or Zero Diameter",
    color = "firebrick",
    size = 6,
    hjust = 0
  ) +
  theme_excel_new() +
  theme(
    legend.position = "none",
    plot.background  = element_rect(fill = "aliceblue", color = NA),
    panel.background = element_rect(fill = "aliceblue", color = NA)
  )

p1/p2
```

```
## Warning: Removed 8718 rows containing missing values or values outside the scale range
## (`geom_ribbon()`).

## Warning: Removed 8718 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



CHECK FOR CORRELATIONS

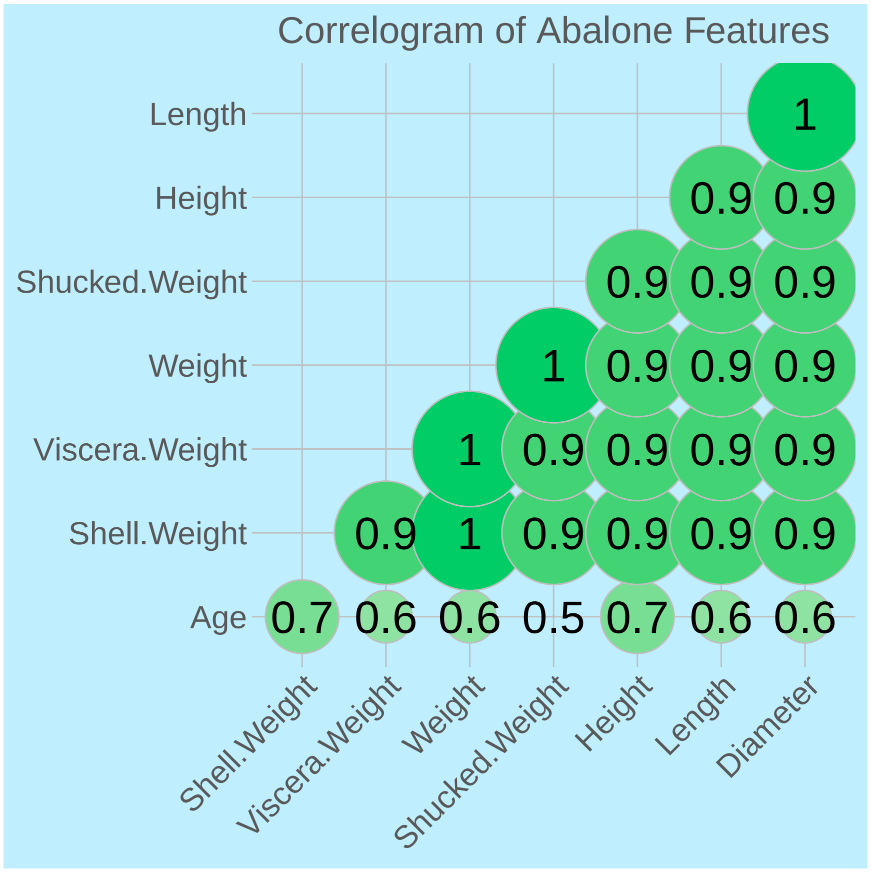
This code yields a correlogram for all numeric features.

All features are strongly correlated with one another, and only moderately correlated with Age. This signals difficulties in producing an accurate predictive model.

```
abalone.corr <- round(
  cor(abalone[,c(Features$Size, Features$Mass, Features$Target)]),
  1
)

abalone.corr %>%
  ggcorrplot(
    type = "lower",
    hc.order = TRUE,
    lab = TRUE,
    lab_size = 6,
    method = "circle",
    colors = c("tomato2", "white", "springgreen3"),
    title = "Correlogram of Abalone Features",
    ggtheme = theme_excel_new
  ) +
  scale_size(range = c(0, 20)) +   # << enlarge circles
  theme(
    legend.position = "none",
    plot.background  = element_rect(fill = "lightblue1", color = NA),
    panel.background = element_rect(fill = "lightblue1", color = NA)
  )
```

```
## Warning: ‘aes_string()‘ was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with ‘aes()‘.
## i See also ‘vignette("ggplot2-in-packages")‘ for more information.
## i The deprecated feature was likely used in the ggcorrplot package.
##   Please report the issue at <https://github.com/kassambara/ggcorrplot/issues>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Scale for size is already present.
## Adding another scale for size, which will replace the existing scale.
```



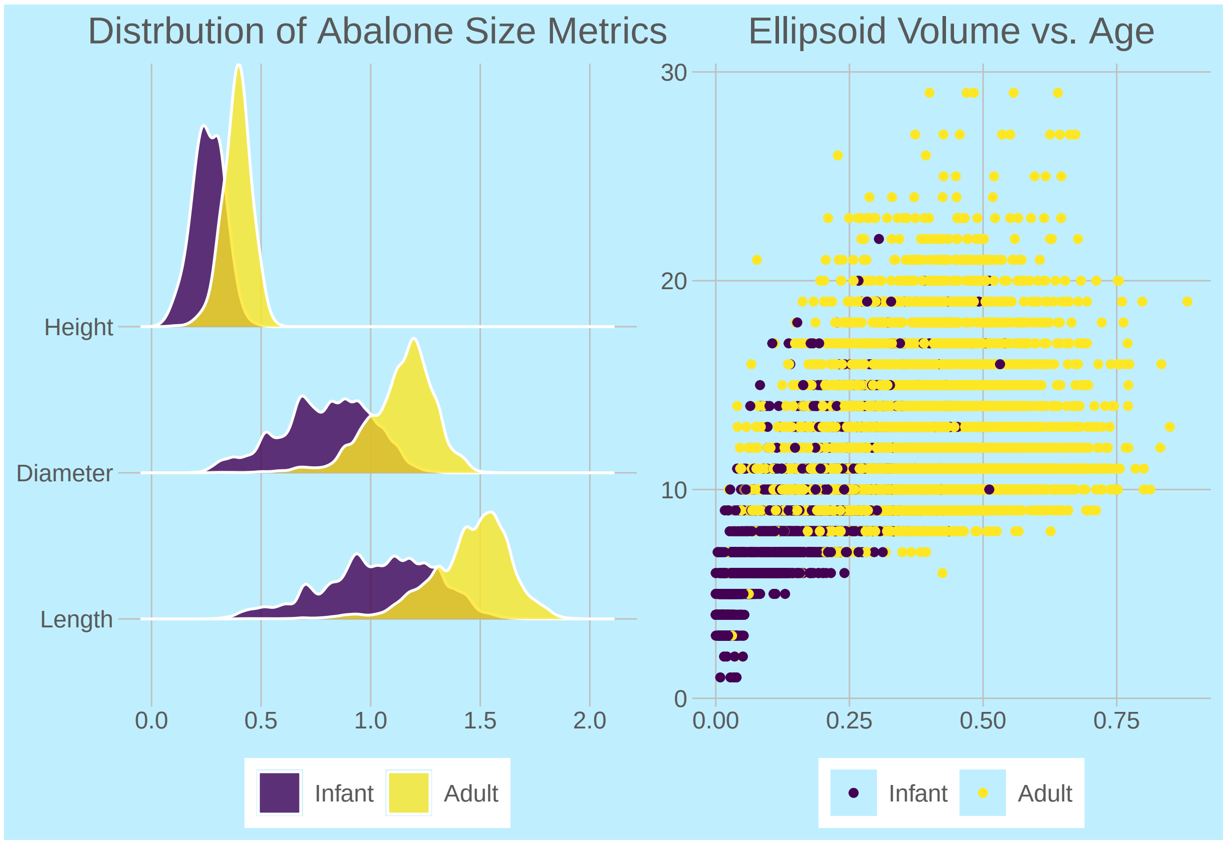
CHECK SIZE AND WEIGHT DISTRIBUTION

Looking at size, the lowest value is high, indicating flatness. Diameter and Length are closer, with most abalone being slightly elongated.

Looking at weight, the the highest mass part is the meat, followed by the shell, and then the viscera.

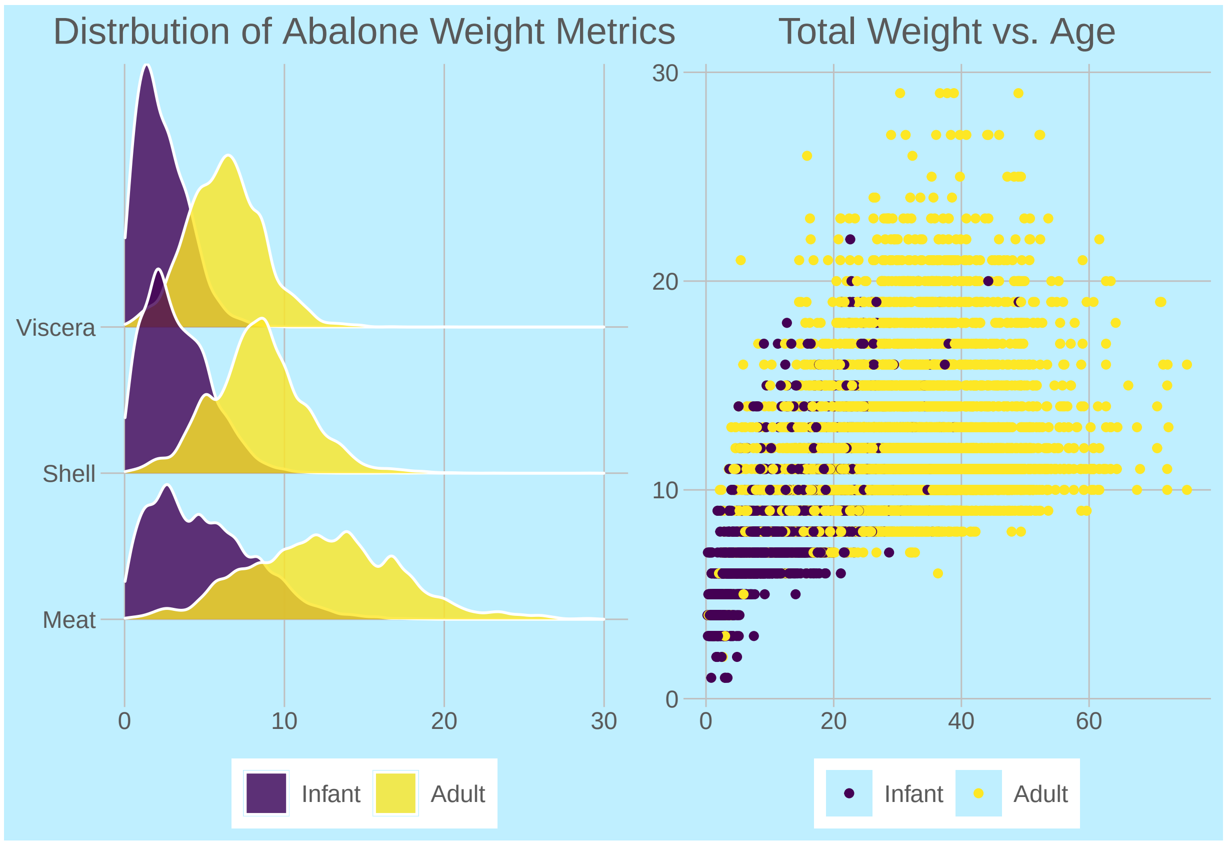
```r
p1 <- abalone %>%
  ggplot(aes(x = Volume, y = Age, color = IsAdult)) +
  geom_point() +
  labs(
    title = "Ellipsoid Volume vs. Age",
    x = expression("Ellipsoid Volume (cm"^3*")"),
    color = NULL
  ) +
  theme_excel_new() +
  theme(
    plot.background  = element_rect(fill = "lightblue1", color = NA),
    panel.background = element_rect(fill = "lightblue1", color = NA)
  )

p2 <- abalone %>%
  pivot_longer(
    cols = c(Height, Diameter, Length),
    names_to = "Measure",
    values_to = "Value"
  ) %>%
  mutate(
    Measure = factor(Measure, levels = c("Length", "Diameter", "Height"))
  ) %>%
  ggplot(aes(x = Value, y = Measure, fill = IsAdult)) +
  geom_density_ridges(alpha = 0.8, color = "white") +
  theme_excel_new() +
  labs (
    title = "Distrbution of Abalone Size Metrics",
    y = "Metric",
    x = "Size (cm)",
    color = NULL
  ) +
  theme(
    plot.background  = element_rect(fill = "lightblue1", color = NA),
    panel.background = element_rect(fill = "lightblue1", color = NA)
  )

p2 + p1
```

```
## Picking joint bandwidth of 0.0232
```



```r
p1 <- abalone %>%
  ggplot(aes(x = Weight, y = Age, color = IsAdult)) +
  geom_point() +
  labs(
    title = "Total Weight vs. Age",
    x = "Weight (grams)",
    color = NULL
  ) +
  theme_excel_new() +
  theme(
    plot.background  = element_rect(fill = "lightblue1", color = NA),
    panel.background = element_rect(fill = "lightblue1", color = NA)
  )

p2 <- abalone %>%
  pivot_longer(
    cols = c(Shucked.Weight, Shell.Weight, Viscera.Weight),
    names_to = "Part",
    values_to = "Value"
  ) %>%
  mutate(
    Part = factor(
      case_when(
        Part == "Shucked.Weight" ~ "Meat",
        Part == "Shell.Weight" ~ "Shell",
        Part == "Viscera.Weight" ~ "Viscera"
      ), levels = c("Meat", "Shell", "Viscera"))
  ) %>%
  ggplot(aes(x = Value, y = Part, fill = IsAdult)) +
  geom_density_ridges(alpha = 0.8, color = "white") +
  theme_excel_new() +
  labs (
    title = "Distrbution of Abalone Weight Metrics",
    x = "Weight (grams)",
    color = NULL
  ) +
  scale_x_continuous(limits = c(0, 30)) +
  theme(
    plot.background  = element_rect(fill = "lightblue1", color = NA),
    panel.background = element_rect(fill = "lightblue1", color = NA)
  )
p2 + p1
```

```
## Picking joint bandwidth of 0.416

## Warning: Removed 36 rows containing non-finite outside the scale range
## (`stat_density_ridges()`).
```



ANALYSIS OF RESIDUALS

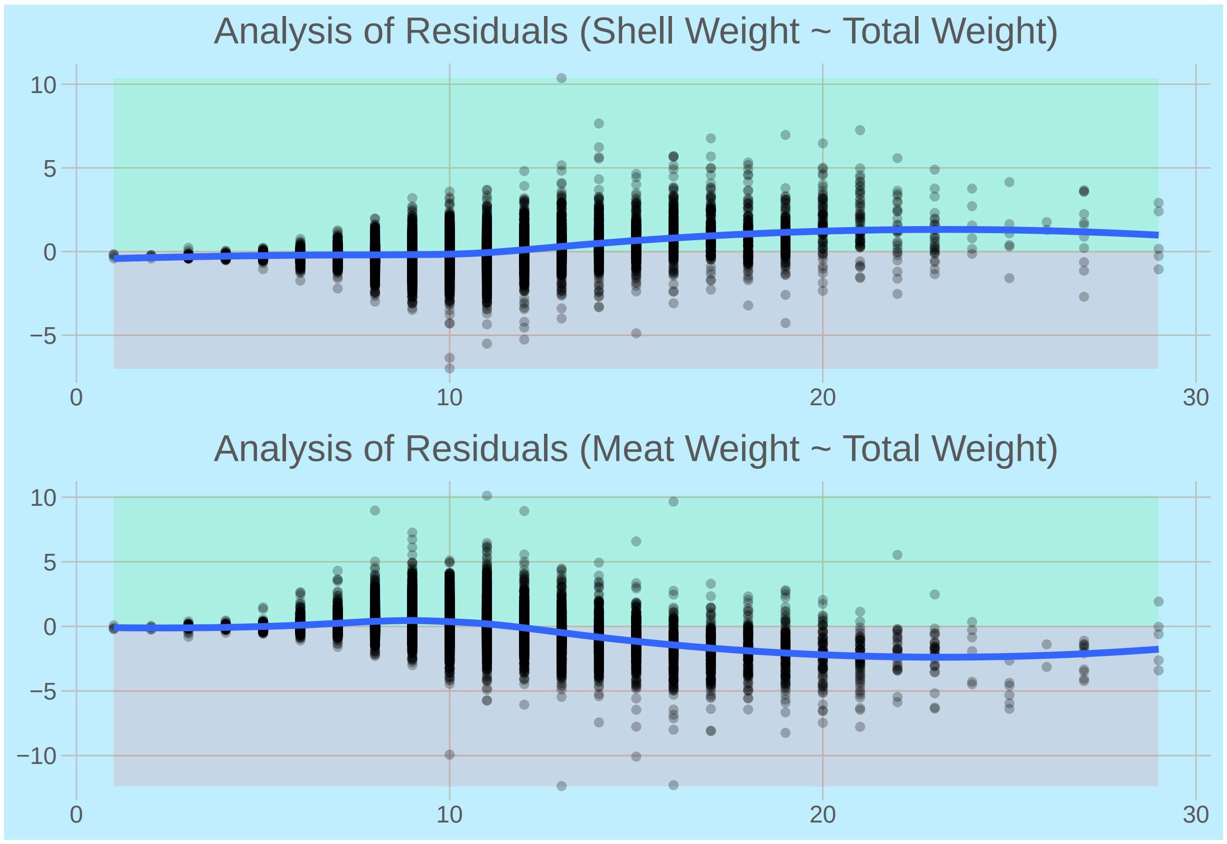
Use residuals to probe the interaction between different variables and Age.

This pair of plots show that as abalone age past 10 years olds, the relative prorportion of meat weight tends to decrease, and shell tends to increase.

```r
analyze_residuals <- function(formula, data, ... ){

  fit <- lm(formula, data = data)
  data <- data %>%
    mutate(
      residual = resid(fit)
    )

  # get x-range for shading
  x_range <- range(data$Age, na.rm = TRUE)
  y_range <- range(data$residual, na.rm = TRUE)

  ggplot(data, aes(x = Age, y = residual)) +
    # === background shading ===
    annotate("rect",
             xmin = x_range[1], xmax = x_range[2],
             ymin = 0, ymax = y_range[2],
             fill = "green", alpha = 0.1) +
    annotate("rect",
             xmin = x_range[1], xmax = x_range[2],
             ymin = y_range[1], ymax = 0,
             fill = "red", alpha = 0.1) +

    # points + smooth
    geom_point(alpha = 0.25) +
    geom_smooth(method = "loess", se = FALSE, linewidth = 1.2) +

    labs(...) +
    theme_excel_new() +
    theme(
      plot.background  = element_rect(fill = "lightblue1", color = NA),
      panel.background = element_rect(fill = "lightblue1", color = NA)
    )
}

p1 <- analyze_residuals(
  Shell.Weight ~ Weight,
  abalone,
  title = "Analysis of Residuals (Shell Weight ~ Total Weight)",
  x = NULL,
  y = "Residual"
  )

p2 <- analyze_residuals(
  Shucked.Weight ~ Weight,
  abalone,
  title = "Analysis of Residuals (Meat Weight ~ Total Weight)",
  x = "Age",
  y = "Residual"
  )
```

```r
p1/p2
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



## MODEL ZOO

We prepare a series of models using all baseline features and various engineered features. The final results are saved to are displayed.

```r
str(abalone)
```

```
## 'data.frame':    15000 obs. of  27 variables:
##  $ id            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Sex           : Factor w/ 3 levels "Female","Infant",..: 3 3 1 3 2 3 3 2 2 1 ...
##  $ Diameter      : num  1.025 0.975 1.375 1.062 0.887 ...
##  $ Length        : num  1.31 1.23 1.74 1.38 1.16 ...
##  $ Height        : num  0.312 0.338 0.5 0.338 0.3 ...
##  $ Weight        : num  21 18.7 54.1 22.7 14 ...
##  $ Shucked.Weight: num  9.89 9.61 19.45 10.9 6.1 ...
##  $ Viscera.Weight: num  4.41 3.77 14.93 5.56 2.61 ...
##  $ Shell.Weight  : num  5.06 4.69 14.22 6.42 3.97 ...
##  $ Age           : int  8 8 11 9 8 14 11 6 5 20 ...
##  $ Volume        : num  0.22 0.211 0.625 0.258 0.162 ...
##  $ IsAdult       : Ord.factor w/ 2 levels "Infant"<"Adult": 2 2 2 2 1 2 2 1 1 2 ...
##  $ Weight.Mean1  : num  6.04 5.54 16.04 7.3 3.98 ...
##  $ Weight.Mean2  : num  8.25 7.51 21.73 9.69 5.46 ...
##  $ Weight.Norm1  : num  11.96 11.34 28.34 13.82 7.73 ...
##  $ Weight.Norm2  : num  24.1 21.9 61 26.5 16 ...
##  $ Size.Norm     : num  1.69 1.6 2.27 1.77 1.49 ...
##  $ Size.Mean     : num  1.38 1.36 1.53 1.41 1.33 ...
##  $ Shell.Ratio   : num  0.241 0.251 0.263 0.283 0.283 ...
##  $ Meat.Ratio    : num  0.472 0.514 0.36 0.481 0.434 ...
##  $ Viscera.Ratio : num  0.21 0.202 0.276 0.245 0.186 ...
##  $ Soft.Ratio    : num  0.682 0.715 0.636 0.726 0.62 ...
##  $ Elongation    : num  1.28 1.26 1.26 1.29 1.31 ...
##  $ Flatness      : num  0.269 0.309 0.323 0.279 0.295 ...
##  $ SurfaceArea   : num  2.08 1.94 3.95 2.28 1.65 ...
##  $ Roundness     : num  0.781 0.796 0.791 0.773 0.763 ...
##  $ Density       : num  0.0105 0.0113 0.0116 0.0114 0.0115 ...
```

```
formulas <- list(
  baseline = Age ~ Sex + Diameter + Length + Height + Weight + Shucked.Weight + Viscera.Weight + Shell.
)

formulas$Weight.Mean1 <- update(formulas$baseline, . ~ . + Weight.Mean1)
formulas$Weight.Mean2 <- update(formulas$baseline, . ~ . + Weight.Mean2)
formulas$Weight.Norm1 <- update(formulas$baseline, . ~ . + Weight.Norm1)
formulas$Weight.Norm2 <- update(formulas$baseline, . ~ . + Weight.Norm2)
formulas$Size.Norm <- update(formulas$baseline, . ~ . + Size.Norm)
formulas$Size.Mean <- update(formulas$baseline, . ~ . + Size.Mean)
formulas$Volume <- update(formulas$baseline, . ~ . + Volume)
formulas$Shell.Ratio <- update(formulas$baseline, . ~ . + Shell.Ratio)
formulas$Meat.Ratio <- update(formulas$baseline, . ~ . + Meat.Ratio)
formulas$Viscera.Ratio <- update(formulas$baseline, . ~ . + Viscera.Ratio)
formulas$Soft.Ratio <- update(formulas$baseline, . ~ . + Soft.Ratio)
formulas$Elongation <- update(formulas$baseline, . ~ . + Elongation)
formulas$Flatness <- update(formulas$baseline, . ~ . + Flatness)
formulas$SurfaceArea <- update(formulas$baseline, . ~ . + SurfaceArea)
formulas$Roundness <- update(formulas$baseline, . ~ . + Roundness)
formulas$Density <- update(formulas$baseline, . ~ . + Density)
formulas$Final1 <- update(formulas$baseline, . ~ . + Soft.Ratio + Weight.Norm1 + Size.Norm)
formulas$Final2 <- update(formulas$baseline, . ~ . + Weight.Norm2 + Volume + SurfaceArea)





results <- eval_circuit(formulas, abalone, excel_path = "results.xlsx")
```

```
## Results written to: results.xlsx
```

```
results %>% select(formula_name, model, mae)
```

```
##     formula_name model      mae
## 1     Soft.Ratio    rf 1.389615
## 2         Final1    rf 1.391122
## 3     Meat.Ratio    rf 1.394211
## 4   Weight.Norm2    rf 1.397993
## 5        Density    rf 1.398905
## 6         Final2    lm 1.398992
## 7       baseline    rf 1.401185
## 8      Roundness    rf 1.401832
## 9   Weight.Norm1    rf 1.402184
## 10     Size.Norm    rf 1.402738
## 11  Weight.Mean1    rf 1.403388
## 12    Elongation    rf 1.403637
## 13  Weight.Mean2    rf 1.404464
## 14     Size.Mean    rf 1.404672
## 15        Final2    rf 1.404745
## 16 Viscera.Ratio    rf 1.404828
## 17        Volume    rf 1.405200
## 18   SurfaceArea    rf 1.406294
## 19  Weight.Norm2    lm 1.406612
## 20      Flatness    rf 1.407100
## 21   Shell.Ratio    rf 1.408511
## 22  Weight.Mean2    lm 1.412326
## 23  Weight.Mean1    lm 1.412638
## 24  Weight.Norm1    lm 1.412816
## 25        Final1    lm 1.416033
## 26        Volume    lm 1.426365
## 27   SurfaceArea    lm 1.427834
## 28       Density    lm 1.432653
## 29     Size.Mean    lm 1.433627
## 30     Roundness    lm 1.438515
## 31     Size.Norm    lm 1.438854
## 32    Elongation    lm 1.439078
## 33 Viscera.Ratio    lm 1.439547
## 34      baseline    lm 1.439702
## 35   Shell.Ratio    lm 1.440525
## 36      Flatness    lm 1.440973
## 37    Soft.Ratio    lm 1.441989
## 38    Meat.Ratio    lm 1.443234
```