Adjust radiomic features: asses best linear model

Carla Casanova

2022-06-21

```
library(datawizard)
library(RadAR)
library(dplyr)
library(ggplot2)
library(ggrepel)
library(RColorBrewer)
library(gridExtra)
```

Load data

Load `RadAR` object with radiomic features:

```
load("/Users/carlacasanovasuarez/Documents/Master Bioinformatics UAB/Prácticas Radiomics/Radiomic featu

rdr_L1
```

```
## class: SummarizedExperiment
## dim: 1232 1778
## metadata(1): extractor
## assays(1): values
## rownames(1232): Elongation.original Flatness.original ...
##   Complexity.wavelet.LLL Strength.wavelet.LLL
## rowData names(4): feature_name image_type feature_description
##   feature_type
## colnames(1778): 23140000005 23140000006 ... 26658003992 26658003997
## colData names(200): filename sample_id ... RACE1.t3 change.fv1
```

Filter by image type, only radiomic features from **original** images will be used:

```
## filter by image types
rdr_filt_original <- filter_by_image_type(rdr = rdr_L1, image_type = c("original"))

rdr_filt_original
```

```
## class: SummarizedExperiment
## dim: 101 1778
## metadata(1): extractor
## assays(1): values
## rownames(101): Elongation.original Flatness.original ...
##   Complexity.original Strength.original
## rowData names(4): feature_name image_type feature_description
##   feature_type
## colnames(1778): 23140000005 23140000006 ... 26658003992 26658003997
## colData names(200): filename sample_id ... RACE1.t3 change.fv1
```

Store radiomic features and standardize data to allow comparisons between different features:

```
# Store radiomic features for original image type (101 features)
table_original <- assay(rdr_filt_original)

# Transpose before scaling in order to scale by features (otherwise scale by
# patients)
table_scaled <- t(scale(t(table_original)))
```

Prepare variables and radiomic table

Store original colnames and rownames:

```
# Store col and row names from the original table
features_original <- rownames(table_scaled)
patients_original <- colnames(table_scaled)
```

Prepare radiomic tables to use `adjust()` function. Variables to be adjusted must be placed as columns and col names starting with integers must be avoided:

```
table_scaled.T <- as.data.frame(t(table_scaled))

# Change invalid column names (white spaces, integers, etc)
names(table_scaled.T)[names(table_scaled.T) == "10Percentile.original"] <- "tenPercentile.original"
names(table_scaled.T)[names(table_scaled.T) == "90Percentile.original"] <- "ninetyPercentile.original"
```

Prepare variables of interest in a data frame:

```
dd.variables <- table_scaled.T %>%
    mutate(Center = as.factor(colData(rdr_filt_original)$CENTREID), Age = colData(rdr_filt_original)$AG
        Smoker = as.factor(colData(rdr_filt_original)$SMOKER), Cough = as.factor(colData(rdr_filt_origi
        Country = as.factor(colData(rdr_filt_original)$COUNTRY), Sex = as.factor(colData(rdr_filt_origi
        GOLDCD = as.factor(colData(rdr_filt_original)$GOLDCD))
```

Now, remove variables with `NaN` values since prediction won't be comparable with original data (complete patients):

```
dd.variables = dd.variables[complete.cases(dd.variables), ]

dim(dd.variables)
```

```
## [1] 1764  108
```

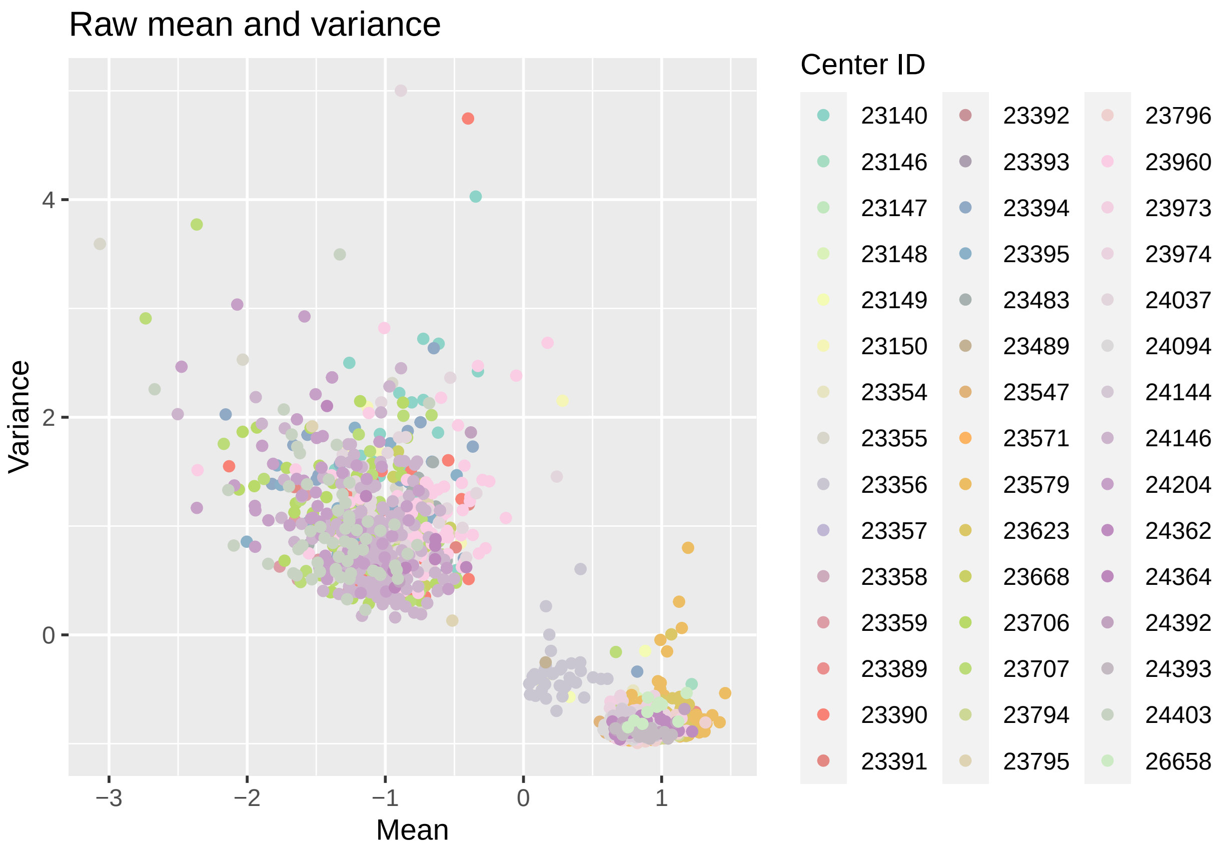
Check data: batch effect and raw data

The following plots clearly show **batch effect** of image data, since three different groups of individuals can be spotted. First, select a palette with the highest number of colors and discrete colors to avoid gradients:

```
# Display palettes display.brewer.all()

# Select discrete palette
mypalette <- brewer.pal(11, "Set3")

# Extend colors by number of levels in Center variable
morecols <- colorRampPalette(mypalette)
n_color <- length(levels(dd.variables$Center))
```

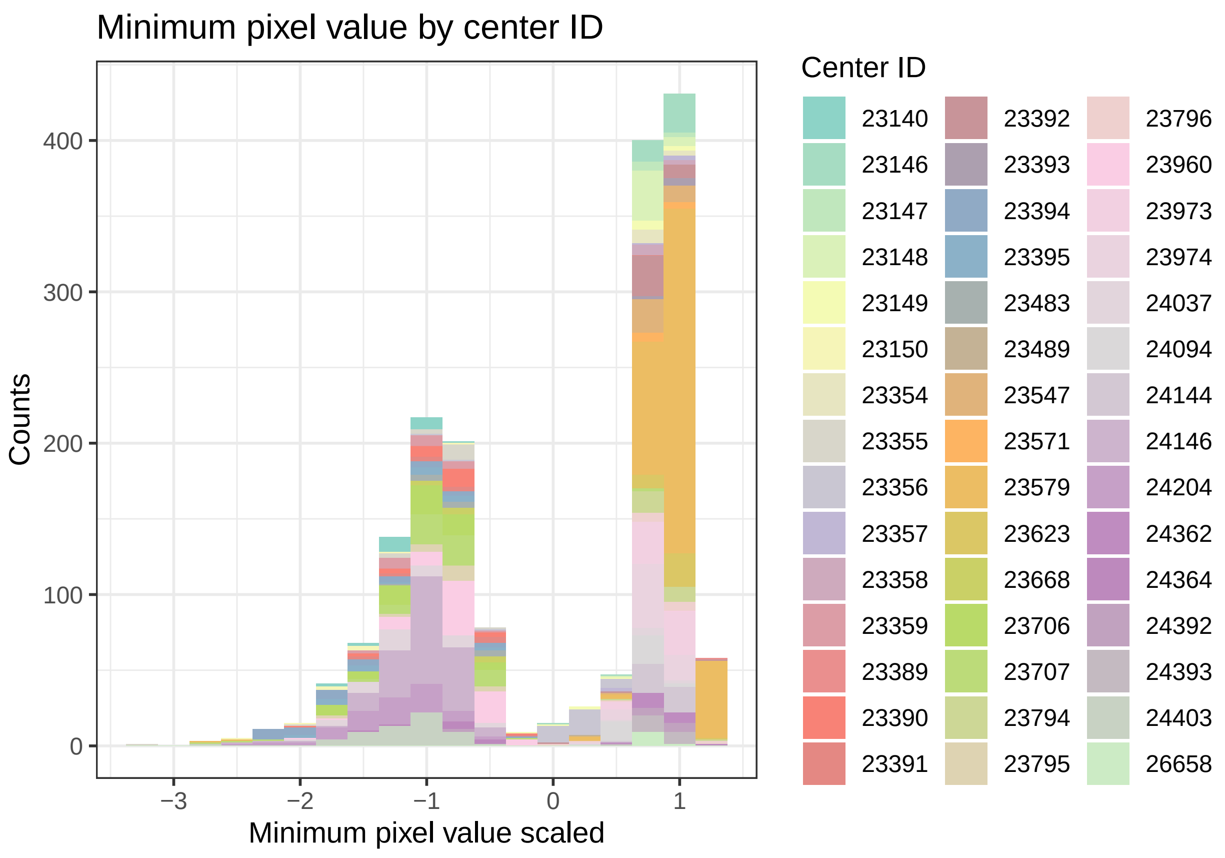
The following plot represents values for `mean` and `variance` coloured by `center ID`, since it is one of the most common variables causing batch effect in radiomics field. Three groups of individuals are clearly differentiated, of which different center composition is also spotted:

```
qplot(dd.variables$Mean.original, dd.variables$Variance.original, colour = dd.variables$Center,
    xlab = "Mean", ylab = "Variance", main = "Raw mean and variance") + labs(colour = "Center ID") +
    scale_colour_manual(values = morecols(n_color))
```



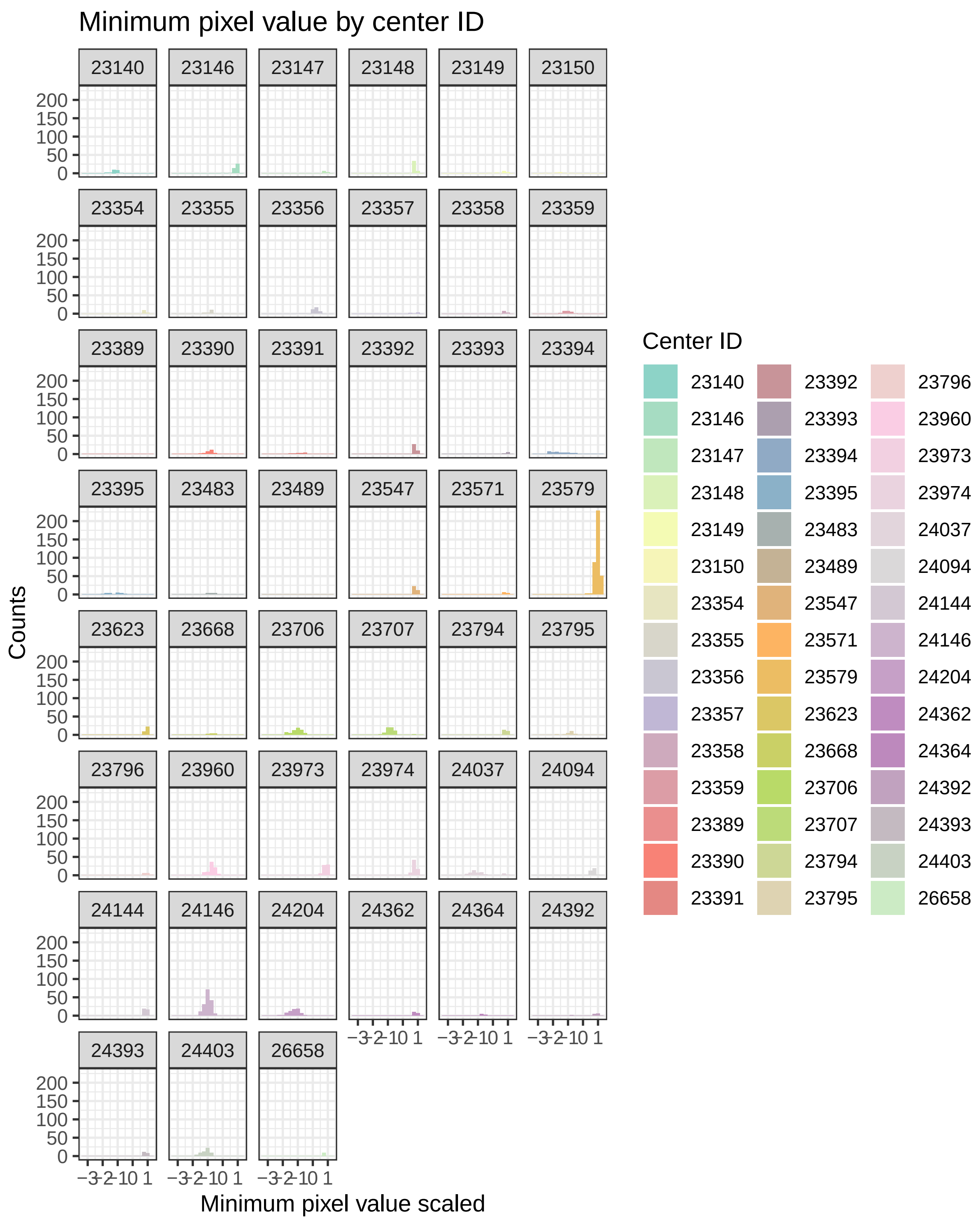
Clearly, `center ID` is affecting radiomic features' values between patients. Let's check data related with the range of pixel values of the images such as `minimum pixel value`. In addition, these values are plotted alongside `center ID` in order to check how much is affecting inter-center variability:

```
ggplot(data = dd.variables) + geom_histogram(aes(x = Minimum.original, fill = Center),
    binwidth = 0.25) + labs(title = "Minimum pixel value by center ID", x = "Minimum pixel value scaled
    y = "Counts", fill = "Center ID") + theme_bw() + scale_fill_manual(values = morecols(n_color))
```



Again, patients displayed different ranges since `minimum pixel values` are different, moreover they are clearly affected by `center ID`. However, see that distribution of `minimum pixel values` is also depending on center:

```
# pdf('minimum distribution by center (raw).pdf')

ggplot(data = dd.variables) + geom_histogram(aes(x = Minimum.original, fill = Center),
    binwidth = 0.25) + labs(title = "Minimum pixel value by center ID", x = "Minimum pixel value scaled
    y = "Counts", fill = "Center ID") + facet_wrap(. ~ Center, ncol = 6) + theme_bw() +
    scale_fill_manual(values = morecols(n_color))
```



```
# dev.off()
```

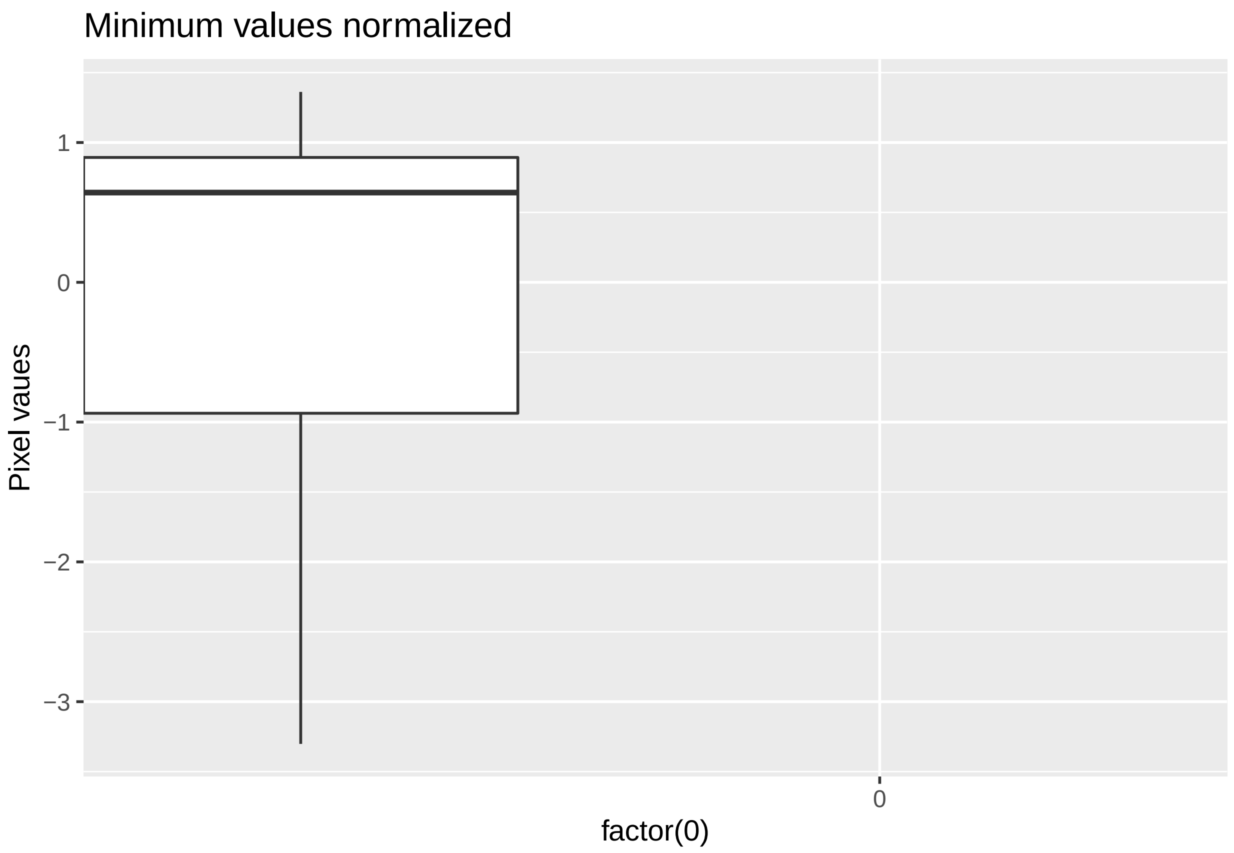
Check for outliers in variables of interest

Regarding ranges of pixel values of CT images, let's check if there are outlier individuals that can affect the analysis:

```r
# Plotting boxplot
is_outlier <- function(x) {
    return(x < quantile(x, 0.25) - 1.5 * IQR(x) | x > quantile(x, 0.75) + 1.5 * IQR(x))
}
```

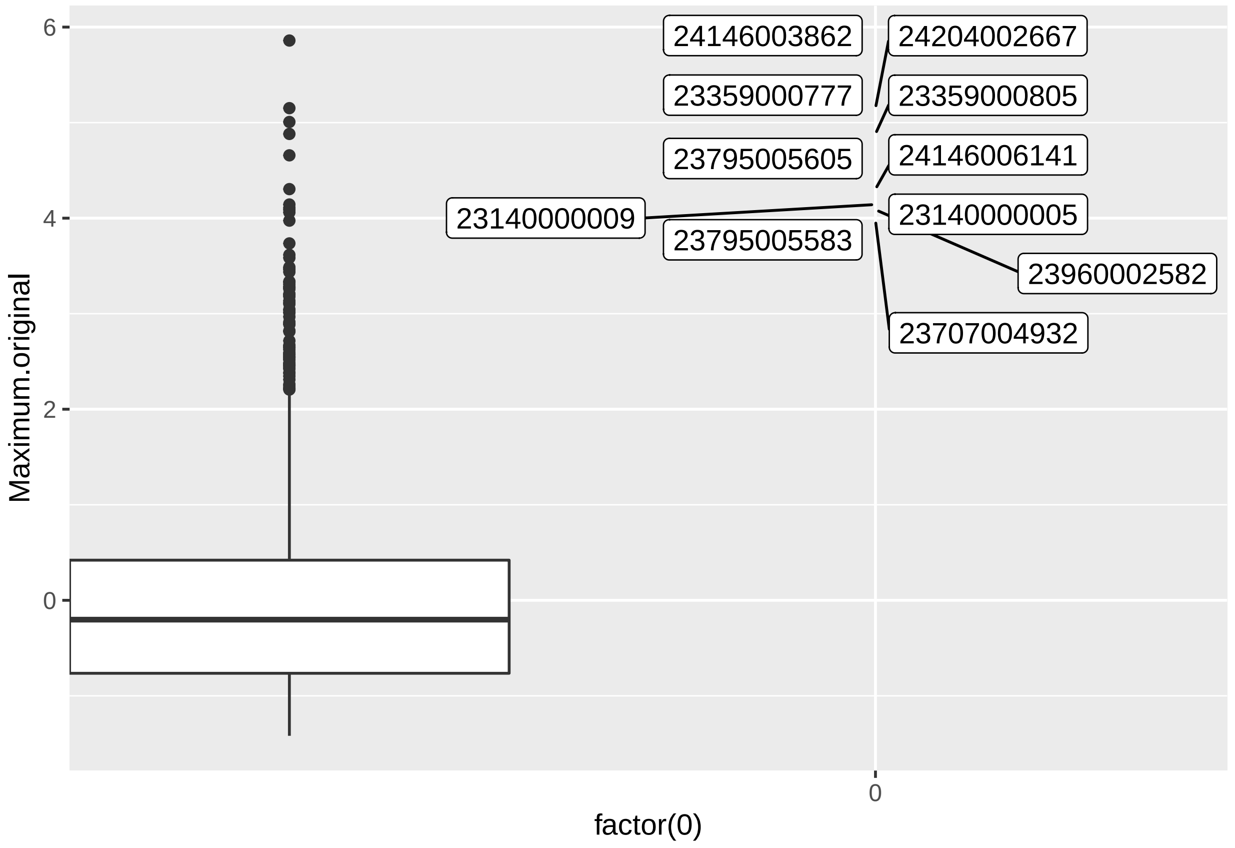
Check if there are outliers and their patient ID for `minimum`:

```r
table_scaled.T %>%
    mutate(outlier = ifelse(is_outlier(Minimum.original), rownames(table_scaled.T),
        as.numeric(NA))) %>%
    ggplot(., aes(y = Minimum.original)) + ylab("Pixel vaues") + ggtitle("Minimum values normalized") +
    geom_boxplot() + geom_text(aes(x = factor(0), label = outlier), na.rm = TRUE,
    hjust = +0.7)
```



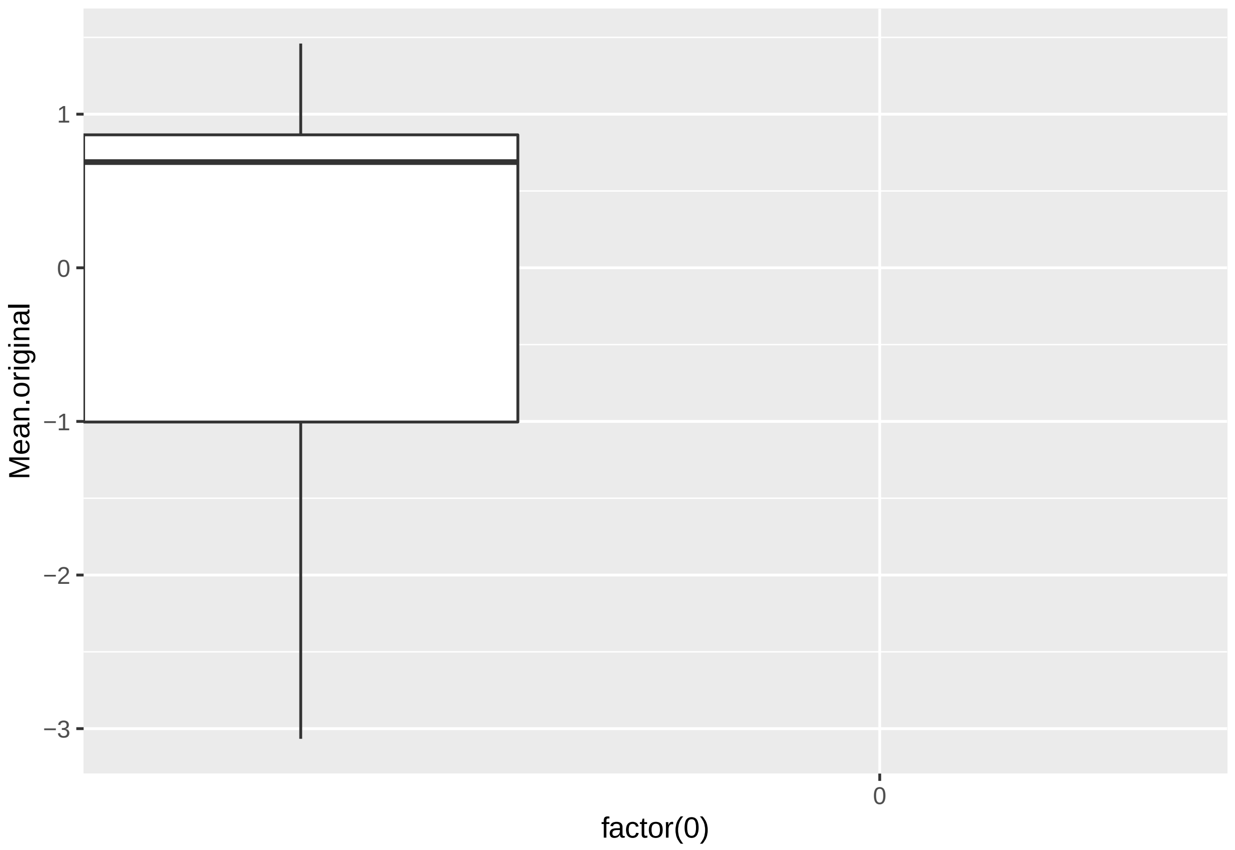
Check if there are outliers and their patient ID for `maximum`:

```r
table_scaled.T %>%
    mutate(outlier = ifelse(is_outlier(Maximum.original), rownames(table_scaled.T),
        as.numeric(NA))) %>%
    ggplot(., aes(y = Maximum.original)) + geom_boxplot() + geom_label_repel(aes(x = factor(0),
    label = outlier), na.rm = TRUE, max.overlaps = 30)
```



Check if there are outliers and their patient ID for `mean`. The `minimum` and `maximum` are vealues representing only one pixel in each case, hence to check if ranges are normal we also need to check the value for most part of the pixels:

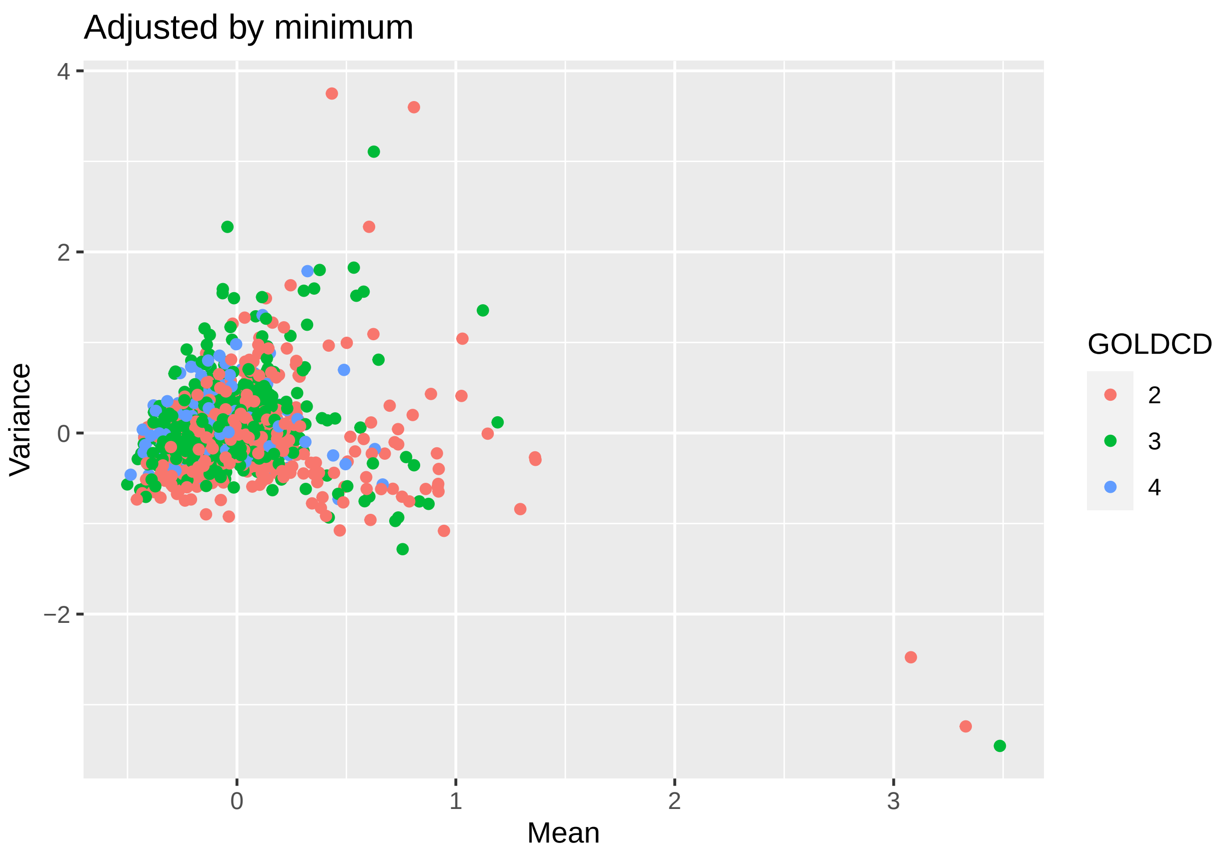
```
table_scaled.T %>%
    mutate(outlier = ifelse(is_outlier(Mean.original), rownames(table_scaled.T),
        as.numeric(NA))) %>%
    ggplot(., aes(y = Mean.original)) + geom_boxplot() + geom_label_repel(aes(x = factor(0),
    label = outlier), na.rm = TRUE, max.overlaps = 30)
```



Adjust

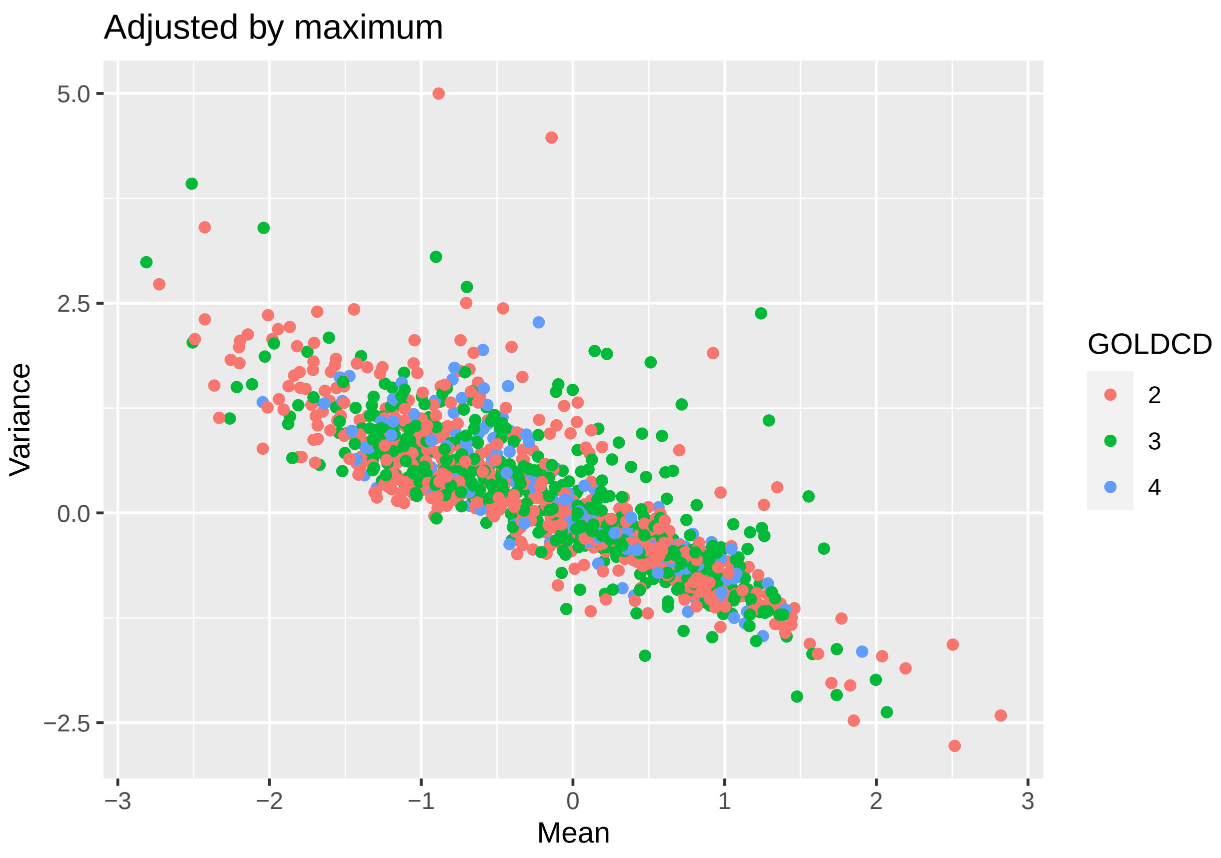
Adjust by `minimum pixel value`:

```
# Adjust all data for the original image type You can adjust just some columns
# with: select = c('Mean.original', 'Variance.original')
clinical_features_min_vox_adj <- adjust(dd.variables, effect = "Minimum.original",
    keep_intercept = TRUE, exclude = c("Age", "Sex", "Smoker", "Cough", "Country",
        "GOLDCD"))

qplot(clinical_features_min_vox_adj[, "Mean.original"], clinical_features_min_vox_adj[,
    "Variance.original"], colour = dd.variables$GOLDCD, xlab = "Mean", ylab = "Variance",
    main = "Adjusted by minimum") + labs(colour = "GOLDCD")
```



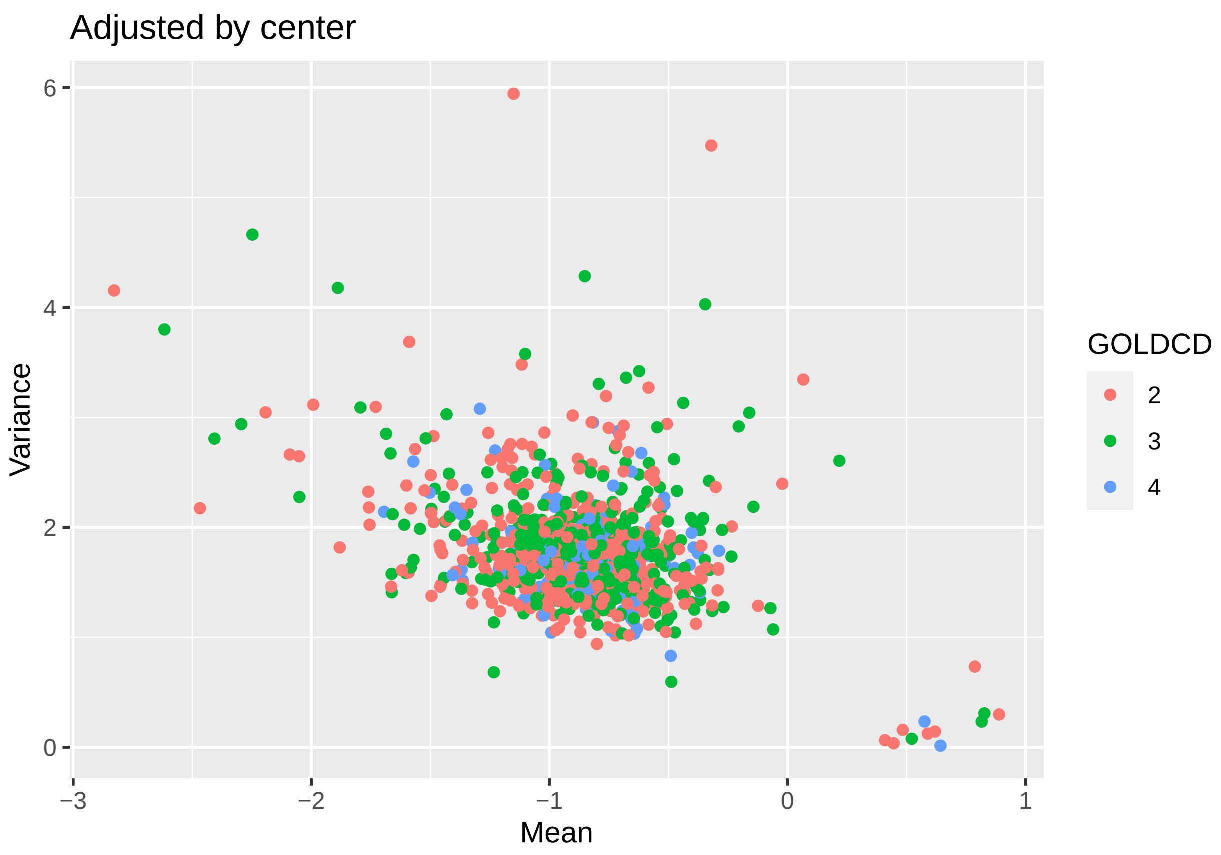
Adjust by `maximum pixel value`:

```
# Adjust all data for the original image type You can adjust just some columns
# with: select = c('Mean.original', 'Variance.original')
clinical_features_max_vox_adj <- adjust(dd.variables, effect = "Maximum.original",
    keep_intercept = TRUE, exclude = c("Age", "Sex", "Smoker", "Cough", "Country",
        "GOLDCD"))

qplot(clinical_features_max_vox_adj[, "Mean.original"], clinical_features_max_vox_adj[,
    "Variance.original"], colour = dd.variables$GOLDCD, xlab = "Mean", ylab = "Variance",
    main = "Adjusted by maximum") + labs(colour = "GOLDCD")
```



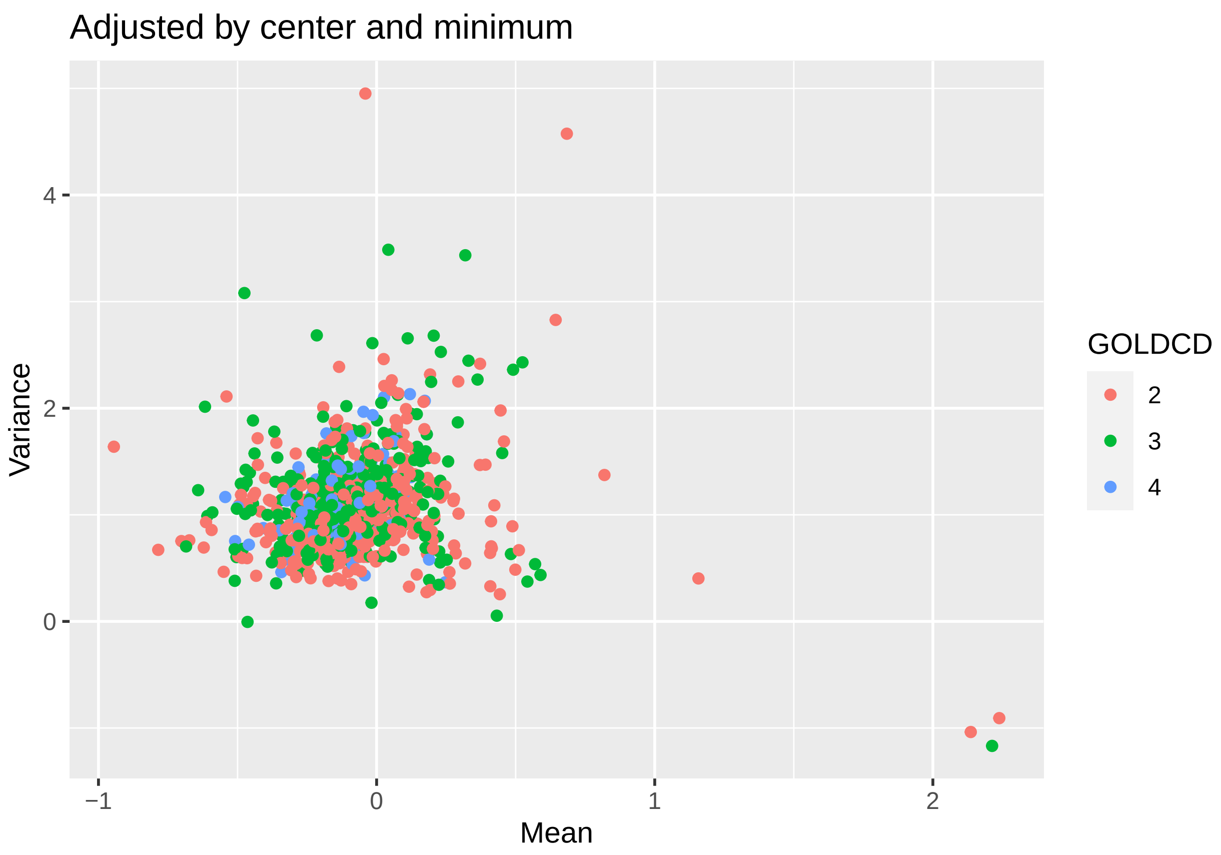
Adjust by `Center`:

```
# Adjust all data for the variable image type You can adjust just some columns
# with: select = c('Mean.original', 'Variance.original')
clinical_features_center_adj <- adjust(dd.variables, effect = "Center", keep_intercept = TRUE,
    exclude = c("Age", "Sex", "Smoker", "Cough", "Country", "GOLDCD"))

qplot(clinical_features_center_adj[, "Mean.original"], clinical_features_center_adj[,
    "Variance.original"], colour = dd.variables$GOLDCD, xlab = "Mean", ylab = "Variance",
    main = "Adjusted by center") + labs(colour = "GOLDCD")
```



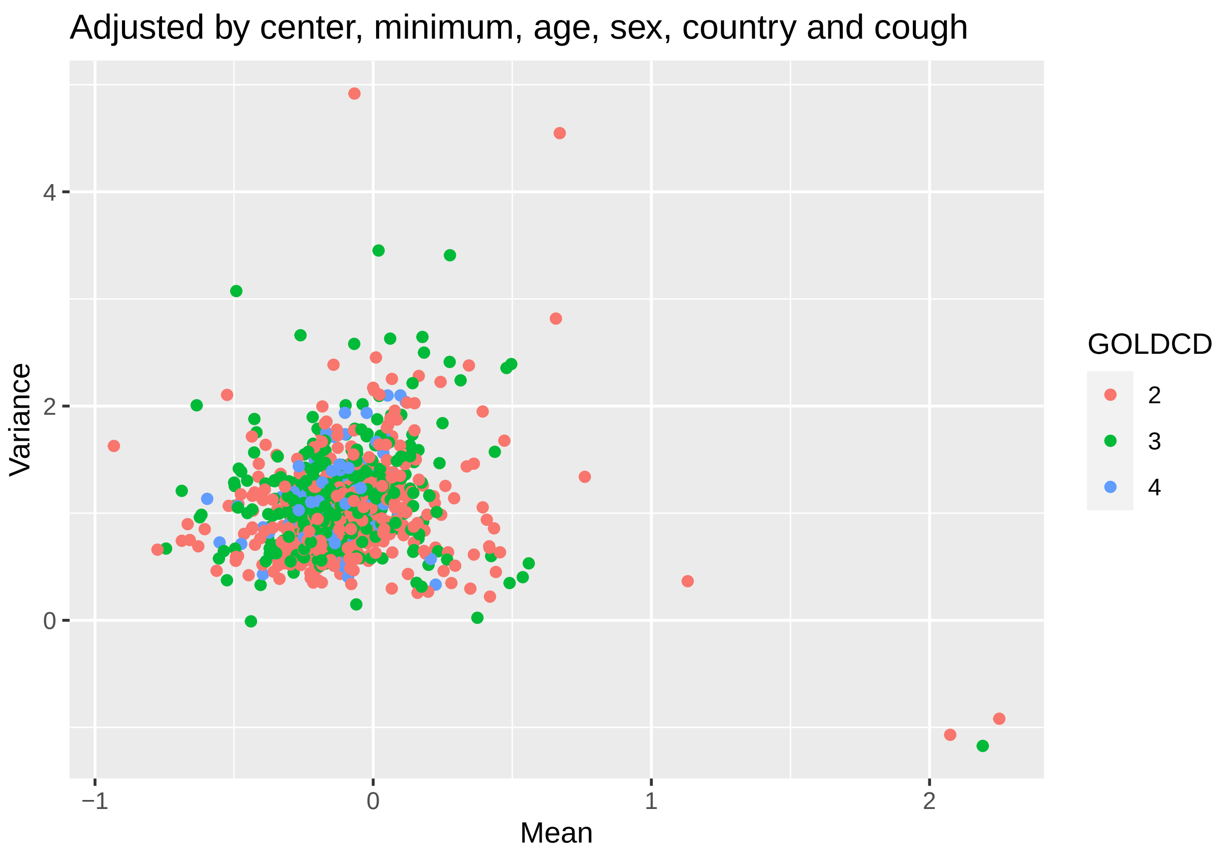
Adjust by `Center` and `minimum`:

```
# Adjust all data for the variable image type You can adjust just some columns
# with: select = c('Mean.original', 'Variance.original')
clinical_features_center_min_adj <- adjust(dd.variables, effect = c("Center", "Minimum.original"),
    keep_intercept = TRUE, exclude = c("Age", "Sex", "Smoker", "Cough", "Country"))

qplot(clinical_features_center_min_adj[, "Mean.original"], clinical_features_center_min_adj[,
    "Variance.original"], colour = dd.variables$GOLDCD, xlab = "Mean", ylab = "Variance",
    main = "Adjusted by center and minimum") + labs(colour = "GOLDCD")
```



Adjust by `Center` and `minimum` plus some other covariates (age, sex, etc):

```
# Adjust all data for the variable image type You can adjust just some columns
# with: select = c('Mean.original', 'Variance.original')
clinical_features_multi_adj <- adjust(dd.variables, effect = c("Center", "Minimum.original",
    "Sex", "Country", "Cough"), keep_intercept = TRUE, exclude = c("Age", "Sex",
    "Smoker", "Cough", "Country"))

qplot(clinical_features_multi_adj[, "Mean.original"], clinical_features_multi_adj[,
    "Variance.original"], colour = dd.variables$GOLDCD, xlab = "Mean", ylab = "Variance",
    main = "Adjusted by center, minimum, age, sex, country and cough") + labs(colour = "GOLDCD")
```



Asses best adjustment

Prepare functions to asses linear models: `RSS` and `RMSE`.

```r
# RSS se ajusta mejor si es más bajo
rss <- function(fitted, actual) {
    sum((fitted - actual)^2)
}

# RMSE se ajusta mejor si es más bajo
rmse <- function(fitted, actual) {
    sqrt(mean((fitted - actual)^2))
}
```

Create the linear regression models:

```r
# By center
simple_model_mean_center <- lm(Mean.original ~ Center, data = dd.variables)

# By minimum
simple_model_mean_min <- lm(Mean.original ~ Minimum.original, data = dd.variables)

# By minimum
simple_model_mean_max <- lm(Mean.original ~ Maximum.original, data = dd.variables)

# By center and minimum
model_mean_center_min <- lm(Mean.original ~ Minimum.original + Center, data = dd.variables)
```

```r
# Multi
model_mean_multi <- lm(Mean.original ~ Minimum.original + Center + Sex + Age + Cough,
    data = dd.variables)
```

Asses all three models:

```r
# RSS
rss(fitted(model_mean_multi), dd.variables$Mean.original)
```

```
## [1] 57.40919
```

```r
rss(fitted(model_mean_center_min), dd.variables$Mean.original)
```

```
## [1] 58.41367
```

```r
rss(fitted(simple_model_mean_center), dd.variables$Mean.original)
```

```
## [1] 141.3399
```

```r
rss(fitted(simple_model_mean_min), dd.variables$Mean.original)
```

```
## [1] 110.5897
```

```r
rss(fitted(simple_model_mean_max), dd.variables$Mean.original)
```

```
## [1] 1130.321
```

```r
# RMSE
rmse(fitted(model_mean_multi), dd.variables$Mean.original)
```

```
## [1] 0.180402
```

```r
rmse(fitted(model_mean_center_min), dd.variables$Mean.original)
```

```
## [1] 0.1819734
```

```r
rmse(fitted(simple_model_mean_center), dd.variables$Mean.original)
```

```
## [1] 0.283063
```

```r
rmse(fitted(simple_model_mean_min), dd.variables$Mean.original)
```

```
## [1] 0.2503849
```

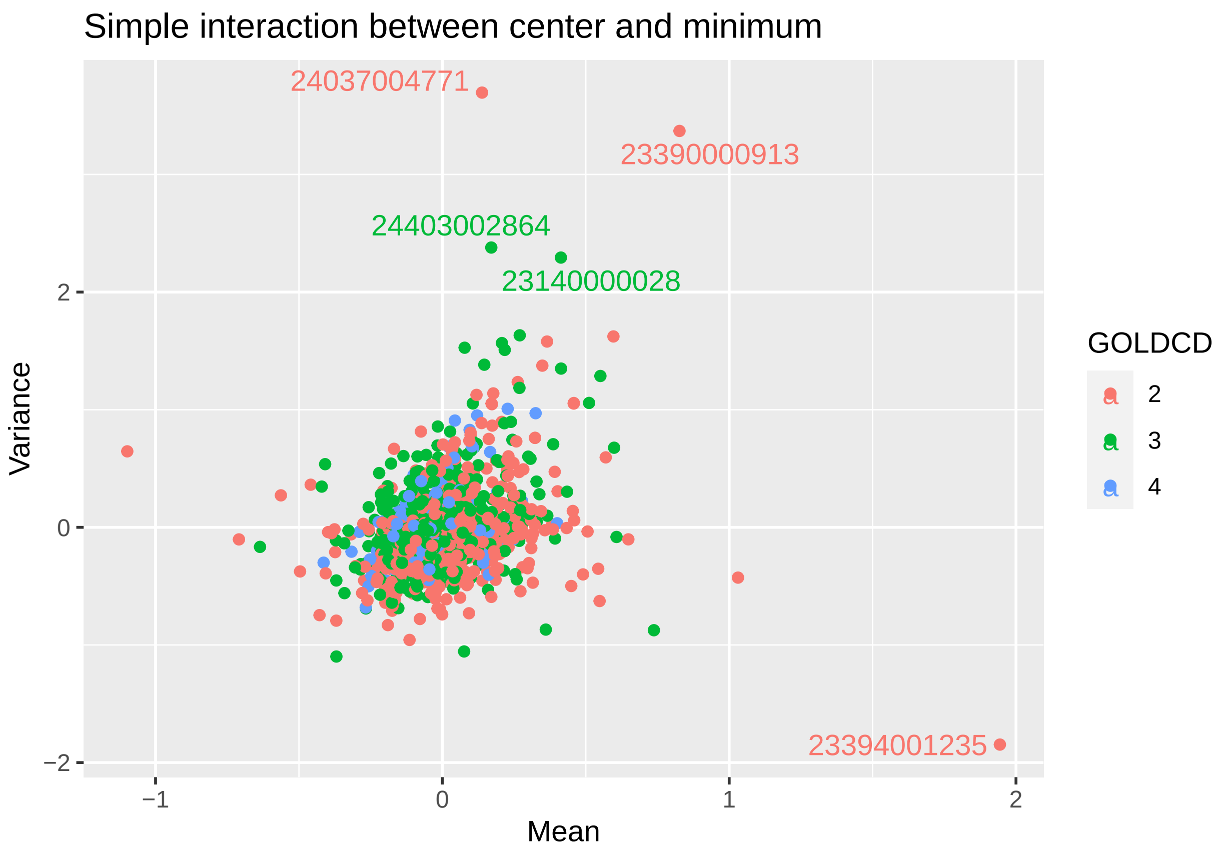
```r
rmse(fitted(simple_model_mean_max), dd.variables$Mean.original)
```

```
## [1] 0.800482
```

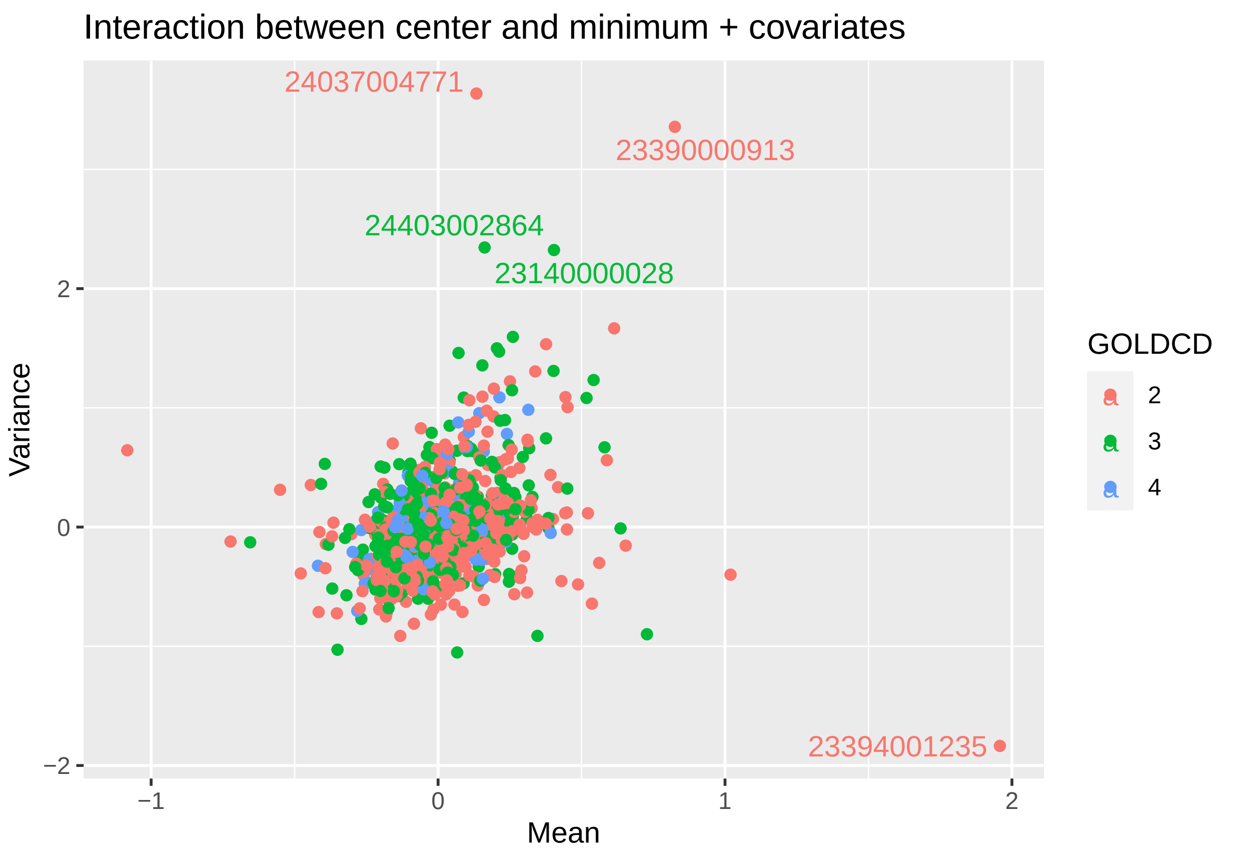
See the difference between addition and interaction of covariates:

```r
# By center and minimum when interacting
model_mean_center_min_int <- lm(Mean.original ~ Minimum.original * Center, data = dd.variables)
model_variance_center_min_int <- lm(Variance.original ~ Minimum.original * Center,
    data = dd.variables)

# By center and minimum when interacting + multi
multi_model_mean_center_min_int <- lm(Mean.original ~ Age + Sex + Country + Minimum.original *
    Center, data = dd.variables)
multi_model_variance_center_min_int <- lm(Variance.original ~ Age + Sex + Country +
    Minimum.original * Center, data = dd.variables)

# Store residuals of simple interaction model to make plots
```

```
Results1 <- data.frame(row.names = rownames(dd.variables), Mean = residuals(model_mean_center_min_int),
    Variance = residuals(model_variance_center_min_int))

# Store residuals of covariates + interaction model to make plots
Results2 <- data.frame(row.names = rownames(dd.variables), Mean = residuals(multi_model_mean_center_min
    Variance = residuals(multi_model_variance_center_min_int))

# Set labels only for patients highly variable
set.label1 <- ifelse(Results1[, "Mean"] > 1.5 | Results1[, "Variance"] > 2, rownames(Results1),
    "")
set.label2 <- ifelse(Results2[, "Mean"] > 1.5 | Results2[, "Variance"] > 2, rownames(Results2),
    "")

# Plot results
qplot(Results1[, "Mean"], Results1[, "Variance"], colour = dd.variables$GOLDCD, xlab = "Mean",
    ylab = "Variance", main = "Simple interaction between center and minimum") +
    labs(colour = "GOLDCD") + geom_text_repel(aes(label = set.label1))
```



```
qplot(Results2[, "Mean"], Results2[, "Variance"], colour = dd.variables$GOLDCD, xlab = "Mean",
    ylab = "Variance", main = "Interaction between center and minimum + covariates") +
    labs(colour = "GOLDCD") + geom_text_repel(aes(label = set.label2))
```



```
# RSS
rss(fitted(model_mean_center_min_int), dd.variables$Mean.original)
```

```
## [1] 40.12806
```

```
rss(fitted(multi_model_mean_center_min_int), dd.variables$Mean.original)
```

```
## [1] 39.84061
```

```
# RMSE
rmse(fitted(model_mean_center_min_int), dd.variables$Mean.original)
```

```
## [1] 0.1508255
```

```
rmse(fitted(multi_model_mean_center_min_int), dd.variables$Mean.original)
```

```
## [1] 0.1502843
```

Conclusion

`Minimum pixel value` adjust data better than `Center` (RSS equal to 110.5897 and 141.3399 respectively), but when both are used as covariates the model fits better (RSS equal to 58.41367). When using multivariate model (adding `Age`, `Sex`, `Country`, `Cough`), the model is slightly improved (RSS equal to 57.40919). In addition, `maximum pixel value` does not fit the model well and it might be possible due to the presence of several outliers (RSS equal to 1130.321).

Nevertheless, the best improvement is when `Center` and `minimum` interact with each other (RSS equal to 40.12806). Adding more covariates to the previous interaction also improves slightly this model (RSS equal to 39.84061).

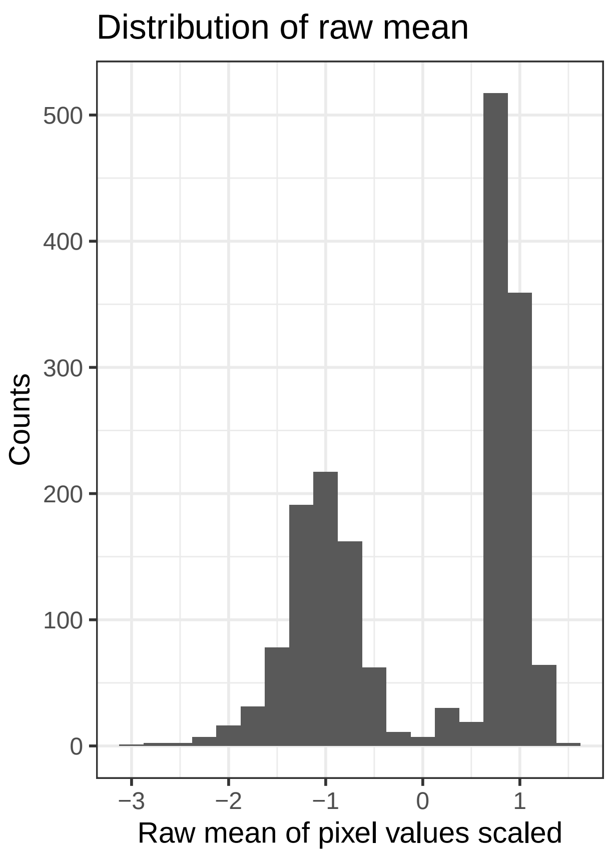
Use the best model to adjust radiomic features

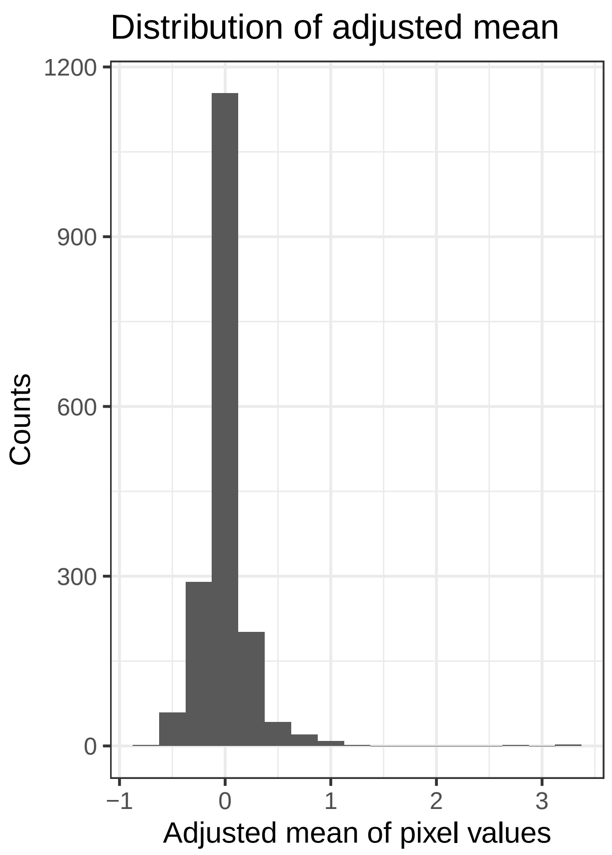
As previously seen, the best model is `minimum` and `center ID` interaction with covariates. Nevertheless, since some covariates have `NaN` and the model is also a good fit without using them, only `minimum` and `center ID` interaction has been considered:

```r
# Data frame with variables of interest
variables <- cbind(Center = colData(rdr_filt_original)$CENTREID, Minimum = table_scaled.T$Minimum.origi
# Empty data frame to store residuals once the model is adjusted
mat_res <- DataFrame(row.names = patients_original)

# Store residuals iterating by columns (by features)
mat_res <- apply(table_scaled.T, 2, function(x) {
    ex <- data.frame(feature = x)
    ex <- cbind(ex, variables)
    residuals(lm(feature ~ Minimum * Center, data = ex, na.action = na.exclude))
})
```

Compare data distribution of raw and adjusted features:

```r
# Raw Mean
plt1 <- ggplot(data = table_scaled.T) + geom_histogram(aes(x = Mean.original), binwidth = 0.25) +
    labs(title = "Distribution of raw mean", x = "Raw mean of pixel values scaled",
        y = "Counts") + theme_bw()

# Adjusted Mean
plt2 <- ggplot(data = as.data.frame(mat_res)) + geom_histogram(aes(x = Mean.original),
    binwidth = 0.25) + labs(title = "Distribution of adjusted mean", x = "Adjusted mean of pixel values
    y = "Counts") + theme_bw()

grid.arrange(plt1, plt2, ncol = 2)
```





Check normally distributed features by using **Shapiro test**:

```
# Apply shapiro test iterating by rows (radiomic features)
results.adj <- apply(mat_res, 2, shapiro.test)
results.raw <- apply(table_scaled.T, 2, shapiro.test)

# Check which features are normally distributed
normal.adj <- sapply(results.adj, function(x) x$p.value > 0.05)
normal.raw <- sapply(results.raw, function(x) x$p.value > 0.05)

# Check how many variables are normally distributed
table(normal.adj)
```

```
## normal.adj
## FALSE  TRUE
##   100     1
```

```
table(normal.raw)
```

```
## normal.raw
## FALSE  TRUE
##   100     1
```

As can be spotted, radiomic features are not normally distributed neither in raw data nor in adjusted data. Nevertheless, histogram of mean pixel values was improved even though there are some individuals highly dispersed. Thefore, data has been transformed from **bimodal distribution** to **right skewed**, nevertheless highly variable individuals can be removed for the analysis.

Exporting the best model to rdr object

Before storing the adjusted data back to the `rdr` object, prepare the table for having the same format than the original data:

```
# Restore original col names (remember that some changes were previously made)
colnames(mat_res) <- features_original

# Transpose the dataframe to place features as rows and individuals as columns
assay_adjusted_original <- as.data.frame(t(mat_res))
```

Filter from the data set **highly variable individuals** (the most dispersed):

```
variable.ind <- intersect(set.label1, colnames(assay_adjusted_original))

# Filter radiomic assay and rdr object
assay_adjusted_original <- assay_adjusted_original[, !(colnames(assay_adjusted_original) %in%
    variable.ind)]
rdr_filt_original <- rdr_filt_original[, !(colnames(rdr_filt_original) %in% variable.ind)]

# Check individuals have the same order in both objects ordered
identical(colnames(rdr_filt_original), colnames(assay_adjusted_original))
```

```
## [1] TRUE
```

```
# Check rdr
rdr_filt_original
```

```
## class: SummarizedExperiment
## dim: 101 1773
## metadata(1): extractor
## assays(1): values
## rownames(101): Elongation.original Flatness.original ...
##   Complexity.original Strength.original
## rowData names(4): feature_name image_type feature_description
##   feature_type
## colnames(1773): 23140000005 23140000006 ... 26658003992 26658003997
## colData names(200): filename sample_id ... RACE1.t3 change.fv1
```

Finally, add a new `assay` to `rdr` object and save it:

```
# This code allows adding assays to rdr object
assays(rdr_filt_original)$adjusted_min_int_center <- assay_adjusted_original
```

Save the object:

```
save(rdr_filt_original, file = "/Users/carlacasanovasuarez/Documents/Master Bioinformatics UAB/Prácticas
```