Bootstrap task for resampling chapter 5: classification of wet and drenched days and bootstrapped ROC of its performance

1. **Fit logistic regression models** Analogously to the task of chapter 4, fit a logistic regression model to mornings with measurable precipitation (`location$wet`) using even mornings (i.e. mornings of days 2, 4, 6, 8, . . .) for training and assign it to `trainLocation`,
   and use odd mornings (of days 1, 3, 5, 7, . . .) for testing (assign to `locationTest`). You can find even and odd days with the modulo function `%% 2`. Replace “location” with the location assigned to you.

Repeat for mornings with at least 12 mm of precipitation (`ibk$drenched`). Remark: since you have to perform 2 very similar tasks that differ only in the precipitation threshold, you might want to write function(s) to achieve them. This is optional but would be good programming practice. For a good and simple introduction on how to write functions in R, see https://discdown.org/rprogramming/functions.html.

2. **Plot the ROC curve for the *training* data set** with 90% confidence interval bands obtained by bootstrap, one figure for each of wet and drenched mornings. Use the package `ROCit` to compute the ROC curve. Comment on the width of the confidence interval. The vignette of the package gives a good overview of how to use it: https://cran.r-project.org/web/packages/ROCit/vignettes/my-vignette.html. First fit a roc-object with `rocEmp <- rocit()` using the empirical method. `class` are the data you used for classifying the data, i.e. the observed wet (or drenched) vs. dry mornings; `score` is your predicted model (here: predictions on the training data). Then compute the 90% confidence interval with `roc90 <- ciROC()` using 100 bootstrap samples. Finally, use `plot(roc90)` with any line/color options of your choice. Note that not all functions of `ROCit` can handle zoo-objects (the format our data set is in). If you get an error message you can try again by stripping the date/time information from the offending variable using `coredata(offendingVariable)`. You may, however, ignore warning messages.

3. **Add the test data ROC curve** Add the ROC curve without confidence intervals for the prediction for the test data period to the previous plot. Why does it (not) fall within the 90% confidence intervals?

4. **ROC with probability thresholds** The ROCit package also provides the probability thresholds used to plot the ROC curve. Its name in the package is “Cutoff”. Exploit that to plot the ROC curve color-coded by the probability threshold values. Use a color palette from the `colorspace` package (https://cran.r-project.org/web/packages/colorspace/vignettes/colorspace.html). Use a sequential multi-hue palette with 10 color bands: `colRoc <- sequential_hcl(10, palette = "Hawai")`. Use `rocNp <- rocit(..., method = "non")` to create a roc-object fit with the non-parametric method. Add a color variable to the `rocNp` object: `rocNp$col`. The color is determined by the threshold probability contained in `rocNp$Cutoff`. For example, elements of `rocNp` with probability thresholds between 0 and 0.1 would be assigned the first color band of `colRoc`; thresholds between 0.9 and 1 to the last color band. Note, though, that the ROCit package also uses probability threshold values outside of [0, 1], which you will have to treat appropriately. Use `plot(..., type = p, pch = 19, cex = 0.5)` to plot the color-coded ROC-curve (and investigate what `type` and `pch` options do).