Tasks for chapter 4 – classification

Task 1: Classification: forecast of wet/dry mornings

This task will look at mornings (06 - 12 UTC) that had measurable precipitation > 0.1 mm.

Again, use the SYNOP observations and forecast data from ECMWF with a forecast horizon of 36 hours for the "location" assigned to your dyad, contained in `location_obs_ECMWF_2009-2022.rds`. In the following text, replace `location` with the name of the one assigned to you, e.g. `ibk`. The rds-file contains several forecast parameters for which acronyms, description and units are given in `ECMWF_selected_parameters.pdf`.

1. **Data preparation:** Add a column with a binary variable `location$wet` to your `location` zoo-object; it should be 1 for a wet morning (`location$rr6hObs > 0.1`) and 0 for a dry morning. Use `ifelse()`. You might want to remove NA values.

2. **Base rate:** What is the base rate of wet mornings? "Base rate" is the probability of an event without using any additional information. In this case it is the fraction of 6-h mornings which had *observed* measureable precipitation. Exploiting that we gave a wet morning a value of 1 and a dry morning a value of 0, the number of wet mornings can be simply computed with `sum(location$wet)` and the total number of mornings with `length(location$wet)`. Alternatively, `sum(location$rr6hObs>0)` will also work for the number of wet mornings since a logical inquiry results in 1 or 0.

3. **Splitting data set into training and testing parts:** Subset your data to use 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`.

4. **Classify:** Logistic regression belongs to the family of generalized linear regression models, which can be fit in R with `glm`. To select logistic regression you have to use it with the option `family = binomial`. Fit a logistic regression with only 1 predictor variable, the ECMWF 6-h precip sum `location$tp6h`, to the training data and assign the model to the variable `glm.fit`. Remember that specifying a model in R is simple with the tilde `~` character: `response ~ predictor1 + predictor2 ...`. The response variable in this case is `rr6hObs`.

5. **Goodness of fit:** `summary()` of the model fit on training data. The z-statistic is the equivalent of the t-statistic in linear regression. It is the coefficient divided by its standard error (cf. James2021, p 136). Note, that the (absolute value of the) z-statistic is the number of standard deviations away from the mean. An absolute value of at least 1.96 indicates a p-value of 0.05: only in 5% of the cases could the null hypothesis be true by chance. Is there an assocation between observed wet mornings and their forecast (i.e. can the null hypothesis be rejected)? Further explanation of the output of `summary(glm.fit)`: *deviance* generalizes RSS (residual sum of squares) to a broader class of models (it is the maximized log-likelihood times -2). The smaller the deviance, the better the fit. The *null deviance* shows how well the response is predicted by the model with only an intercept. The *residual deviance* shows how well the response is predicted by the model when predictor(s) are included. *Degrees of freedom*: number of observations minus number of predictors.

6. **Predict:** Use the model `glm.fit` to make predictions on the training data set (assign to `predictTrain`) and test data set (`predictTest`), respectively using `predict()`, which takes as arguments the model (`glm.fit`, the data (`trainLocation` or `testLocation`) and `type = "response"`.

7. **Confusion matrix:** Compute the confusion matrix separately for predictions on the training data set and the test data set, respectively, using a probability p=0.5 as threshold. What does the confusion matrix tell you about the types of mistakes that the logistic regression model makes in this case? Use `table()` to compute the confusion matrix. The option `deparse.level = 2` will provide a complete labeling of the confusion matrix and its columns.

8. **ROC curve:** Compute the ROC for training and test data and plot both lines in one figure (cf. James2021, Fig. 4.8). Use the true positive rate on the y-axis and the false positive rate on the x-axis. See tables 4.6 and 4.7 in James2021 and - more exhaustive - the Wikipedia entry for confusion matrix for all the different metrics (and their multiple names) that can be computed from the confusion matrix (also known as "contingency table"). For example, *true positive rate* (== true positives divided by observed positives) is also called *sensitivity*, and *true negative rate* (== true negatives divided by observed negatives) is also called *specificity*. *false_positive_rate* is `1 - specificity`. What are the values for the area under the curve (AUC)? How much worse is the performance on the test data compared to the training data? Think of an end-user for whom a high true positive rate is crucial, who is willing to tolerate a false positive rate of 20%, and who refuses to get a probability but wants a yes/no of "wet" forecast. What would be

the optimal probability to convert the probabilistic forecast into a binary one for this end-user? Many packages exist in R to plot ROC-curves. 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 vs. dry mornings; `score` is your predicted model (here: predictions on the training data and test data, respectively). Since `rocit` cannot handle the time information contained in the zoo-object, hand over just the data without the date information with `coredata`, e.g. `coredata(predictTrain)`. Also suppress plotting the legend with the option `legend = FALSE`. Use `names(rocTest)` to see which variables the roc-object contains. Then plot the ROC curve for the test data with `lines(rocTest$TPR ~ rocTest$FPR, col = "red")`. Notice that `~` indicates that you are plotting a function when you use this command! Finally add a legend with `legend()` and the options `horiz = TRUE, lwd = 4` and the AUC-values (with `text()` and `paste()` to concatenate "AUC:" and the AUC value rounded to 3 digits: `round(rocTrain$AUC,3)`).

9. **Precision-recall (PR) curve:** Other information from the confusion matrix can be extracted and plotted, too. If the base rates of the 2 classes (wet and dry in our case) would differ strongly (say a hundred times more events in one class than the other), a precision-recall curve is better suited to display the skill of the model. "Precision" (or "positive prediction value") is the number of true positives normalized by the sum of true positives and false positives. "Recall" is another expression for "sensitivity" or "true positive rate". Plot a figure with precision as a function of recall for the model predictions for training and test data, respectively. Use `type = "p", pch = 19, cex = 0.15` as plot options and add the test data PR curve with `points()`.
10. **Multi-predictor classification:** Use your meteorological knowledge (or intuition if from another Master's program) to select 2 additional predictors. Repeat subtasks 4 - 7 to compute confusion matrices for training and test data and compare results with the model using only 1 predictor. Compute and plot ROC and AUC and compare them to the 1-predictor model.

Task 2: Atmospheric/cryospheric examples for using Poisson regression

Think of an example from atmospheric or cryospheric science with *count* data (e.g. number of people using a bike offered by a bike sharing company as described in section 4.6.2 of James2021). What is the response and what are potential predictors? If you are from another Master's program, use an example from your field.