IML Term project paper

Julia Palorinne, Pyry Silomaa, Sara Sippola

23.12.2022

Introduction

In this project we trained a classifier on a data set of atmospheric measurements. The task is to predict whether new particle formation (NPF) happens or not on a given day based on the atmospheric data. We ran several different kinds of models, assessed through accuracy and perplexity calculations which of these models performed the best, and in this report we discuss the process and the results of our analyses. Our main reference throughout the report is the course book (James, Witten, Hastie, & Tibshirani, 2021), and if a reference is not specified when discussing machine learning theory, the reader can safely assume that we rely on (James et al., 2021).

Preprocessing of the data

Initial data analysis

The goal of this assignment is to predict the behavior of a multinomially distributed variable "class4", which describes new particle formation events on specific observation days. The variable is treated as a multinomial variable divided into four separate groups which are according to the article "Formation and growth of fresh atmospheric aerosols: eight years of aerosol size distribution data from SMEAR II, Hyytiälä, Finland" (Maso et al., 2005): no new particle formation (referred to in the dataset as "nonevent"), very clear and strong particle formation events (referred to in the dataset as class "Ia"), other particle formation events where the growth and formation rate could be determined with a sufficiently good confidence level (referred to in the dataset as class "Ib") and particle formation events where growth and formation rates could either not be defined or there was a possibility that the assessed rates were not sufficiently accurate. For the most part, we are observing a simpler version of this variable instead in the form of a binomial variable, let's refer to that as "class2", which is simply divided into no new particle formation events happening and new particle formation events happening.

Our dataset includes 464 observations of this variable as well as corresponding observations to a collection of 103 other variables. These variables include 50 numeric variables which describe the mean measurement related to phenomena such as carbon dioxide concentration, solar radiation and air temperature during a measurement day, 50 numeric variables which describe the corresponding standard deviations to these measurements for those same measurement days, an id listing which runs from 1 to 464, the date of each measurement as well as a logical variable describing whether the other measurements are partly bad. We also notice that some of the 50 types of measurements for which means and standard deviations are calculated are in fact related to the same phenomena but are simply measured at different points. For example, one of the measurements is the amount of carbon dioxide but this is measured at four different heights.

In order to get a preliminary idea of what we are dealing with, let's visualize some of the variables we're observing as well as calculate some very simple measurements of these variables. First, let's download the dataset, define the binomial variable and observe how many of the observation of "class4" fall into specific classes.

```
## [1] "Relative amount of observations in class nonevent: 0.5"
```

```
## [1] "Relative amount of observations in class Ia: 0.0732758620689655"
```

```
## [1] "Relative amount of observations in class Ib: 0.183189655172414"
```

```
## [1] "Relative amount of observations in class II: 0.243534482758621"
```

We observe that half of our observations are days with no new particle formation events, implying that half our observations are days with new particle formation events. Out of the classes related to days with new particle formation events, class I type events are more common than class II type events but class II type events are more common than either class Ia or class Ib type events. And within class type I events, we have more observations of class Ib type events than class Ia type events.

Let's also calculate basic summary statistics for the first few variables. The full summary of the whole dataset is available in appendix A.

```
##        id            date              class4          partlybad
##  Min.   :  1.0   Length:464         Length:464         Mode :logical
##  1st Qu.:116.8   Class :character   Class :character   FALSE:464
##  Median :232.5   Mode  :character   Mode  :character
##  Mean   :232.5
##  3rd Qu.:348.2
##  Max.   :464.0
```

Of particular interest here is that the variable "partlybad" seems to have only observations of type "FALSE". As a result, we can remove it from the dataset as it gives no extra information. We can also remove the variable id, as the order of the observations is not of interest in our analysis. And on top of this, it is unlikely that we will use the specific dates in relation to the observations either, so we will simply shift that information to be the rownames for the data.

Now we are left with our variable of interest, as well as 100 explanatory variables.

The key objective in this project, classifying NPF event and nonevent days correctly, is an example of *supervised learning*: our data set contains the correct classes, and we can use these in the training of our models. With *unsupervised learning* we can look for insights into the data without using the known classes. There are many different types of unsupervised learning, discussed in (James et al., 2021, pp. 497–552), but since our goal is classification, we focus on *cluster analysis*. This means that we look for subgroups of similar observations. In particular, we use K-means clustering to see if we can easily differentiate between event and nonevent days, and between event types Ia, Ib, II and nonevent, without using the given classes except to validate the results. The K-means clustering algorithm starts by assigning each observation to a cluster at random. At every iteration, the algorithm computes a *cluster centroid*, a vector of the variable means for each observation in the kth cluster, and then reassigns each observation to the cluster with the nearest centroid. The nearest centroid is the one to which an observation has the smallest Euclidean distance.

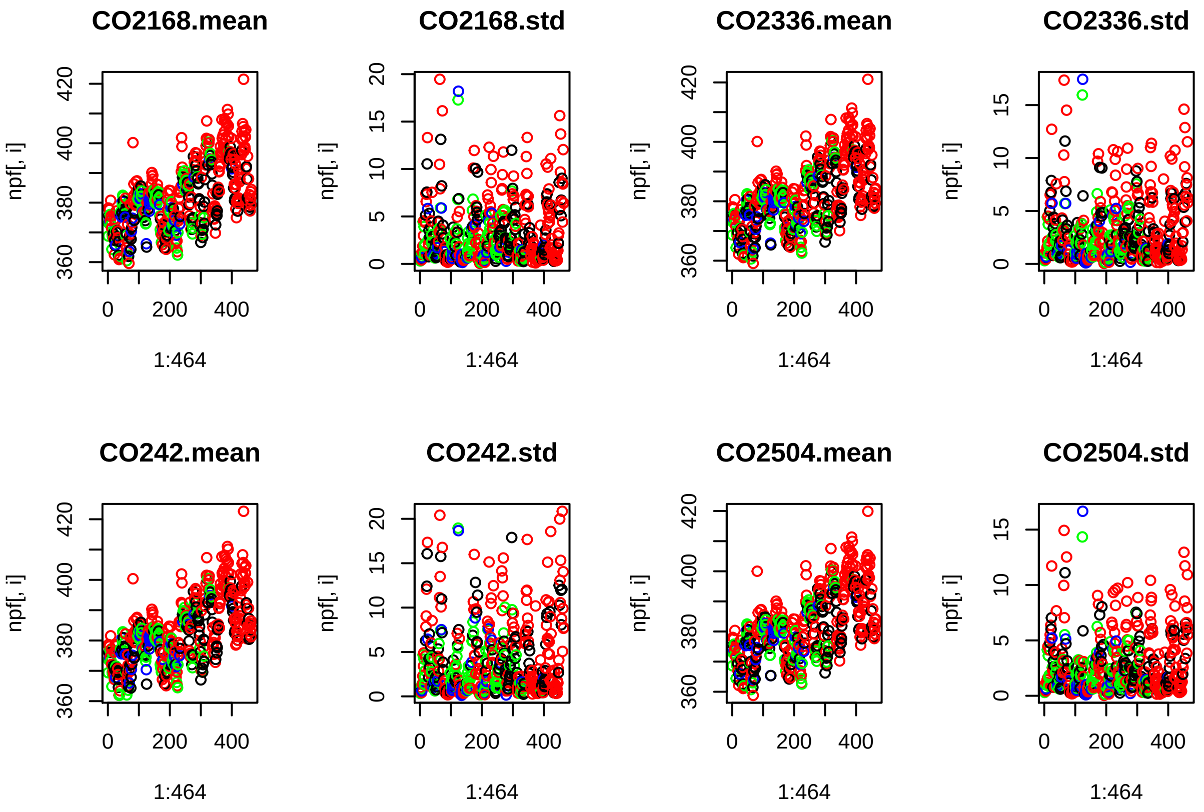
Recall that we have daily means and standard deviations in the dataset. For the purposes of K-means clustering, we restrict ourselves to the means, and because the variables are measured on very different scales, we normalize them to have zero mean and unit variance with the *scale* function. The random cluster assignments tend to have an effect on the end result, so we run the *kmeans* function with 1000 random starting points, and look at the best result. With 2 clusters, the confusion matrix given by the *solve_LSAP* function from the (Hornik & Böhm, 2022) package that matches the clusters with the known classes, is as follows:

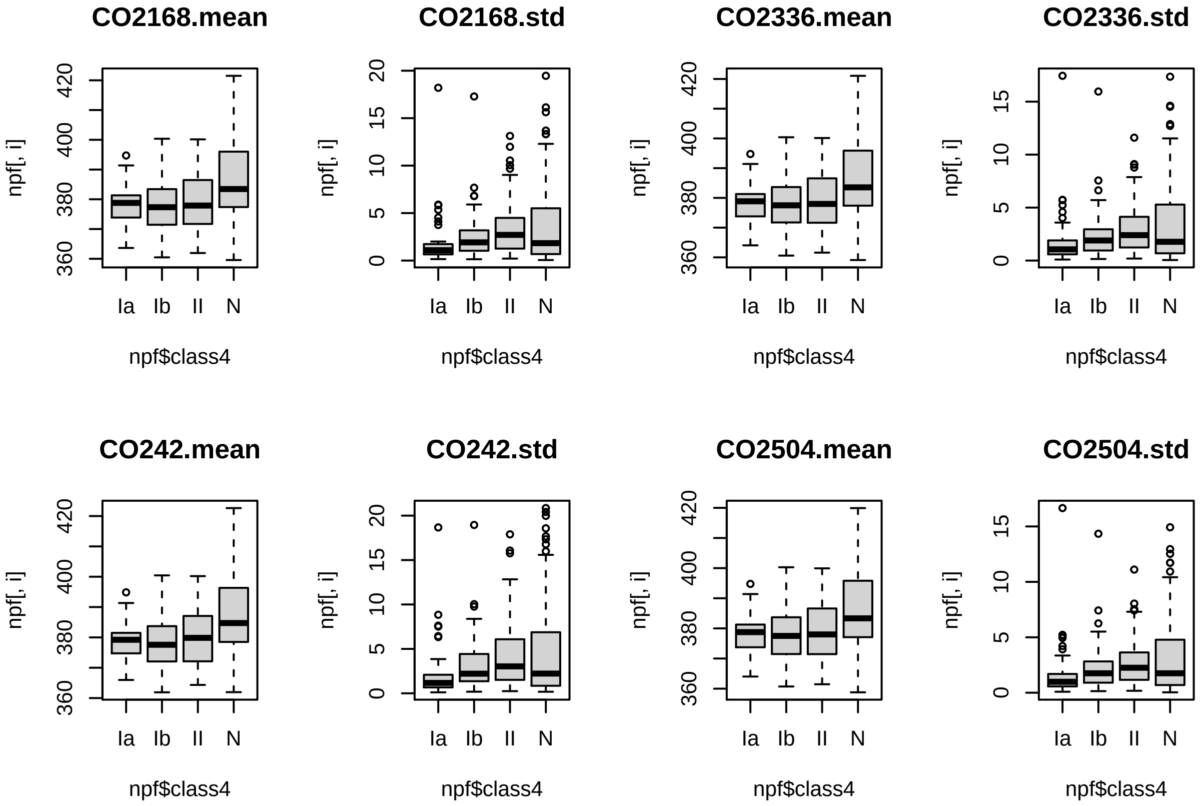
```
##
##                1   2
##   nonevent 134  98
##   event     44 188
```

K-means clustering assigns 178 observations to cluster 1 and 286 to cluster 2, so it does not quite catch the 50-50 division in the dataset. It has assigned 58% of nonevents to class 1, and 81% of events to class 2, which amounts to a total accuracy of 69% if we think of the clustering result as a classifier. With 4 clusters, the accuracy of our cluster-classifier drops down to 46%. The K-means algorithm cannot easily identify the correct event classes.

```
##
##                 4   3   1   2
##   Ia            0  21   4   9
##   Ib            0  56  21   8
##   II            1  52  42  18
##   nonevent     20   4  94 114
```

Let's next visualize with R (R Development Core Team, 2008) some our explanatory variables via boxplots and scatterplots to get a deeper understanding of them. For the sake of simplicity, these visualizations grouped by the phenomena being measured. In order to also preliminarily look at how these variables behave in relation to new particle formation events, we'll also look at said measurements behave in relation to variable "class4". In terms of the boxplots, the boxplots are divided into groups by variable class4, and in terms of the scatterplots, the following colors represent different classes: red represents nonevent days, blue represents days with type Ia NPF events, green represents days with type Ib NPF events and black represents days with type II NPF events:





(Other visualizations omitted from the preliminary report.)

In particular we notice that with several of the variables the boxplots for the nonevent group observations differ more from the observations for days with NPF events than the boxplots for the different NPF event groups both in terms of smaller or higher values but also spanning larger intervals (although this can partially be caused by there being observations within this group). Another notable thing is that the measurements of the same variables at different heights behave very similarly, which imply correlation between them (we take a more detailed look into this in the section *Correlation between parameters*). We also notice a few interesting sets of variables, such as nitrogen monoxide (measurements NO), potential temperature gradient (measurements PTG), rain indicator signal (measurements SWS), sulphur dioxide concentration (measurements SO2) and (measurements CS), which seem to mostly have very small standard deviations. This would imply that the shifts during the day in relation to the corresponding mean observations are on average very small.

Correlation

Correlation between predictors and class

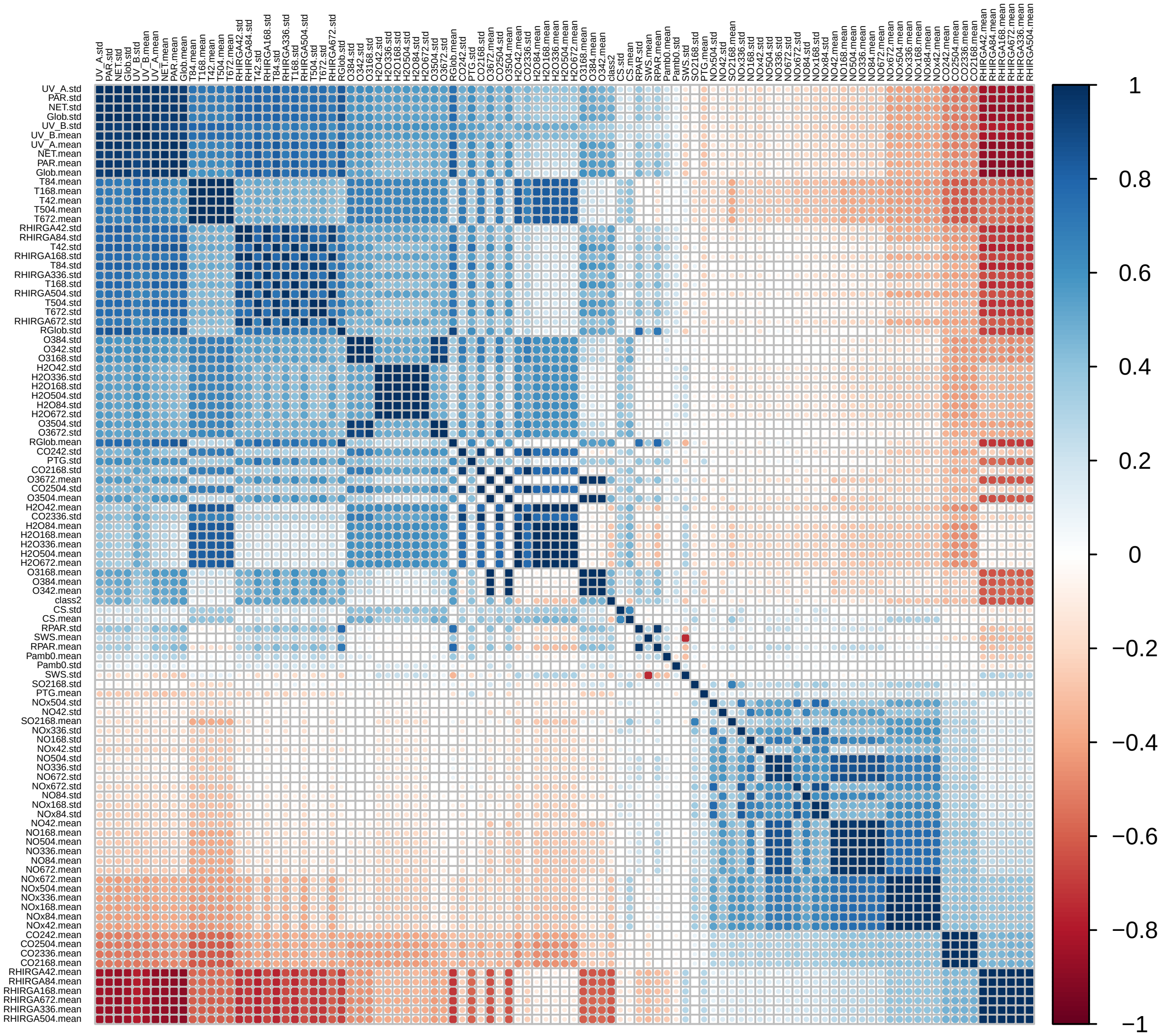
Let's also observe correlations between our explanatory variables by themselves and the variable we're attempting to predict. As there is no natural way of turning the four-class multinomial variable into a numeric form (because the different classes don't have a clear direction in which they rise or fall), let's observe how the explanatory variables correlate with the variable "class" instead. This should help us get a preliminary sense of what explanatory variables might describe our variable of interest the best.

```
##   CO2168.mean    CO2168.std   CO2336.mean    CO2336.std    CO242.mean
##   -0.30626777   -0.09596773   -0.30271045   -0.10806722   -0.32985917
##      CO242.std    CO2504.mean     CO2504.std      Glob.mean       Glob.std
##    -0.08793757    -0.29887003    -0.11393071     0.56928615     0.48236139
##    H2O168.mean     H2O168.std    H2O336.mean     H2O336.std     H2O42.mean
##    -0.27790050     0.05382782    -0.28144835     0.05359948    -0.27131753
##      H2O42.std    H2O504.mean     H2O504.std    H2O672.mean     H2O672.std
##     0.04299127    -0.28349790     0.05559773    -0.28529105     0.04823407
##     H2O84.mean      H2O84.std       NET.mean        NET.std     NO168.mean
##    -0.27456743     0.04349242     0.48609505     0.49417773    -0.12711787
##      NO168.std     NO336.mean      NO336.std      NO42.mean       NO42.std
##    -0.02889094    -0.13899494    -0.06847844    -0.15285335    -0.04893791
##     NO504.mean      NO504.std     NO672.mean      NO672.std      NO84.mean
##    -0.14272836    -0.07816737    -0.15283971    -0.10621825    -0.12969448
##       NO84.std    NOx168.mean     NOx168.std    NOx336.mean     NOx336.std
##    -0.07182797    -0.27466662    -0.08621634    -0.27903932    -0.13286528
##     NOx42.mean      NOx42.std    NOx504.mean     NOx504.std    NOx672.mean
##    -0.27633195    -0.10229014    -0.28264902    -0.10067518    -0.28437084
##     NOx672.std     NOx84.mean      NOx84.std     O3168.mean      O3168.std
##    -0.13807492    -0.27065623    -0.08856140     0.46504624     0.10349253
##      O342.mean       O342.std     O3504.mean      O3504.std     O3672.mean
##     0.46498198     0.11189109     0.46028886     0.06857651     0.45846357
##      O3672.std      O384.mean       O384.std     Pamb0.mean      Pamb0.std
##     0.05833510     0.46599615     0.11090869     0.16284163     0.15887982
##       PAR.mean        PAR.std       PTG.mean        PTG.std     RGlob.mean
##     0.54942511     0.46556287    -0.20133638     0.40030049     0.55973492
##      RGlob.std RHIRGA168.mean  RHIRGA168.std RHIRGA336.mean  RHIRGA336.std
##     0.47701430    -0.62724876     0.53196348    -0.62334360     0.52157036
##  RHIRGA42.mean   RHIRGA42.std RHIRGA504.mean  RHIRGA504.std RHIRGA672.mean
##    -0.63030377     0.54032646    -0.62093390     0.51271466    -0.61695967
##  RHIRGA672.std  RHIRGA84.mean   RHIRGA84.std      RPAR.mean       RPAR.std
##     0.49239697    -0.63023535     0.53490698     0.35123511     0.31087959
##    SO2168.mean     SO2168.std       SWS.mean        SWS.std      T168.mean
##    -0.10607274     0.02183685     0.33076219    -0.26767867     0.09695338
##       T168.std       T42.mean        T42.std      T504.mean       T504.std
##     0.51372072     0.09537137     0.52006333     0.09221095     0.50047378
##      T672.mean       T672.std       T84.mean        T84.std      UV_A.mean
##     0.09038205     0.49176171     0.09791565     0.52080878     0.51594601
##       UV_A.std      UV_B.mean       UV_B.std        CS.mean         CS.std
##     0.44214769     0.39813283     0.34950884    -0.28010884    -0.05254049
##         class2
##     1.00000000
```

Based on this, we might as least preliminarily be interested in measurements of solar radiation (measurements Glob), net radiation (measurements Net), O^3 (measurements O3), potential temperature gradient in C/m (measurements PTG), reflected solar radiation (measurements RGlob), temperature (measurements T)m type-A UV radiation (measurements UV_A) and measurements RHIRGA (which perhaps refer to some kind of relative humidity) as explanatory variables.

Correlation between variables

The correlation between the predictors and the value to be predicted is a very natural starting point when it comes to correlation analysis. It is also possible that the predictors themselves are correlated. Many classifiers are affected by high correlation, i.e. *collinearity* between variables. In regression models, for example, it can be difficult to differentiate between the individual effects of collinear variables on the response. Let's visualize these correlations with R package (Wei & Simko, 2021)



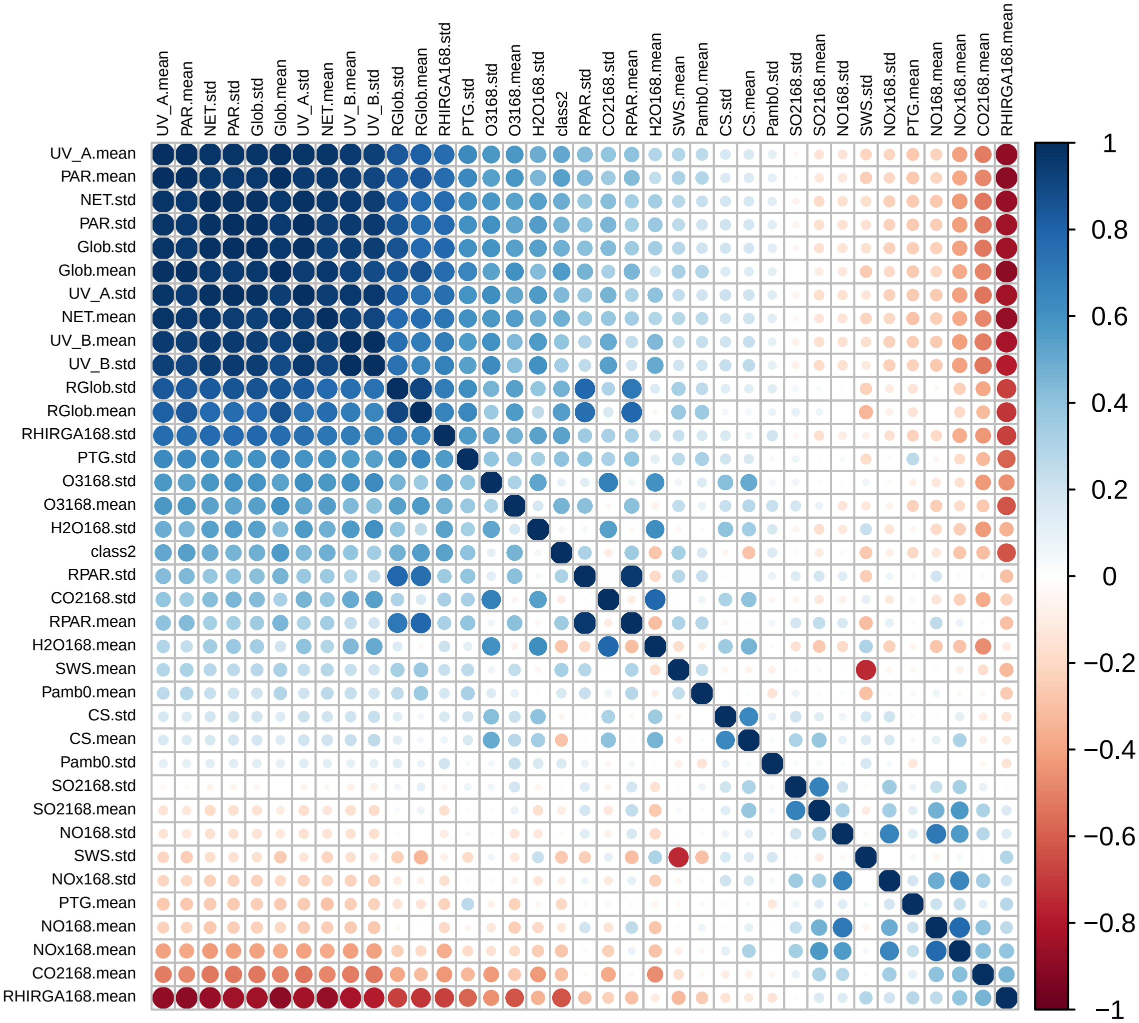
As expected, the correlation plot shows that many of the variables describing the same phenomenon at different heights are correlated, as are variables related to radiation. For many classifiers, highly correlated variables are problematic, and for this reason we would like to drop as many of these columns from the dataset as possible.

We know that many of the variables are measurements of the same thing at different heights. For example, water vapour concentration (H20) has been measured at heights 4.2m, 8.4m, 16.8m, 33.6m, 50.4m and 67.2m. As can be seen from the table below, the correlation coefficients between the daily means of water vapour concentration at different heights are essentially 1.

Table 1: Correlation (H20)

	H2O168.mean	H2O336.mean	H2O42.mean	H2O504.mean	H2O672.mean	H2O84.mean
H2O168.mean	1.0000000	0.9998966	0.9997062	0.9997158	0.9994631	0.9998894
H2O336.mean	0.9998966	1.0000000	0.9993506	0.9999302	0.9997498	0.9996330
H2O42.mean	0.9997062	0.9993506	1.0000000	0.9990202	0.9986416	0.9999330
H2O504.mean	0.9997158	0.9999302	0.9990202	1.0000000	0.9998589	0.9993631
H2O672.mean	0.9994631	0.9997498	0.9986416	0.9998589	1.0000000	0.9990316
H2O84.mean	0.9998894	0.9996330	0.9999330	0.9993631	0.9990316	1.0000000

Hence we discard most of the measurements of H2O and other variables for which we have multiple highly correlating measurements, both the means and the standard deviations. We keep the parameters measured at 16.8 meters as for some parameters, the measurements have only been measured at that height. This rather crude method of discarding data leaves us with 38 unique columns. The figure below describes the correlation coefficients between the remaining variables



The correlation plot for the remaining variables shows us that the remaining highly correlated variables are all radiation-related. When collinearity is reduced, variables are often discarded based on some threshold of the correlation coefficient, e.g. 0.7, but at this point we opt to leave the radiation-related variables be, and reduce collinearity in different ways for the models where it may present problems.

Classifier

Description of considered machine learning approaches

In order to use a model that describes the process that creates the data the best, we attempted to use several different kinds of models to assess the data. We test these models on the original dataset or on a suitably modified version of it. After defining each of these models, we would assess which of the models we used performed best and use said model in our predictions.

Specifically, the two measures we used to assess the quality of each model were its accuracy and perplexity. We defined accuracy similarly as in the course material (James et al., 2021, p. 37) as simply the relative amount of values the model predicted correctly, expressed mathematically for variables of interest y_i and related predictions $\hat{y}_i$ as $\frac{\sum_{i=1}^{n} I(y_i=\hat{y}_i)}{n}$. This measures whether the model defines the labels correctly. All values naturally receieve values within interval $[0, 1]$ and higher values imply better results.

Perplexity instead expresses the relative confidence in those predictions. The models we use give estimated probabilities $\hat{p}(\hat{y}_i = y)$ that the datapoints $\hat{y}_i$ we're predicting receive specific values y. Thus, if the model is descriptive of the phenomena which create the dataset, it would seem reasonable that for the actual realized values y the measure $\hat{p}(\hat{y}_i = y)$ would be relatively high. Thus its logarithmic transformation $\log(\hat{p}(\hat{y}_i = y))$ would also be able to receive high values, and similarly $-\log(\hat{p}(\hat{y}_i = y))$ and $e^{-\log(\hat{p}(\hat{y}_i=y))}$ would receive small values. Thus we can use as a corresponding overall measure for the dataset the measurement $e^{-\frac{\sum_{i=1}^{n} \log(\hat{p}(\hat{y_i}=y_i))}{n}}$ where y_i are realized values of the observations we're attempting to assess through $\hat{y}_i$. As $\hat{p}(y_i = y) \in [0, 1]$ for all $\log(\hat{p}(y_i = y)) \leq 0$ for all y. Thus, all values of our general perplexity measure are within interval $[1, \infty)$ and smaller values are preferred.

However, it is not sufficient to simply define the model on the training dataset and assess accuracy and perplexity on said training dataset as this would woefully inflate the accuracy assessment and deflate the perplexity assessment of the model on a general dataset. We instead attempted to assess accuracy and perplexity through cross-validation methods, specifically leave-one-out cross-validation (loo-cv). The idea behind loo-cv according to the course material (James et al., 2021, pp. 200–202) is that for each observation in the training dataset, a model is trained based on all other observations in the dataset. This new model is then used to predict the training dataset value that was left out of defining the model. With this method, we receive based on observations of explanatory variables both predicted values and estimates of probabilities for specific events for each measurement in a way that didn't include the observations in both defining the model training and assessing its quality. Thus, we can use these measures in our accuracy and perplexity calculations to better assess how the model using all the observations in the training dataset behaves on a completely separate test dataset.

Next, let's discuss some of the models we considered

Dummy model

As our first model, we simply attempted to predict the values through a dummy model, which predicts all values into the more common of the two classes. However, as half of the observations are nonevent days and half of the observations are event days, accuracy would only be 50% no matter which the model would choose. As a result, the dummy model is not sufficient to use in this problem

Logistic regression

According to the course material (James et al., 2021, pp. 133–137), Logistic regression models can be used for binomial variables as well as for multinomial variables under specific assumptions. The idea of logistic regression in a binary case is that our variable of interest Y_i are binomially distributed with parameter $q \in [0,1]$ representing probability that Y_i receieves one of the values. Now, with explanatory variables X_{ij} we assume that these X_{ij} influence the parameter p_i for each observation, meaning $p(Y_i|X_i) = q(X_i)^{Y_i}(1 - q(X_i))^{1-Y_i}$ is a binomial probability with parameter $q(X_i)$ being of form $\sigma(\sum_j \lambda_j X_{ij})$ for values of individual explanatory variables X_{ij} and coefficients λ_j and link function $\sigma : \mathbb{R} \to [0,1]$. Thus we can estimate a logistic regression by finding parameters λ_j that maximize the likelihood function $\prod p(Y_i|X_i)$.

Notable benefits of the logistic regression model include that according to the article “On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes”(Ng & Jordan, 2001) as a discriminative model it eventually reaches a lower asymptotic variance than comparable generative models. On top of this, according to the course material [book, p. 387], compared to for example comparable support vector classifiers, on average logistic regression models fare better under situations where the classes are not easily separable by explanatory variables.

However, possible shortcomings of the logistic regression model include assumptions of linearity in relation to the explanatory variables and the parameter (which we could attempt to correct but with such a large collection of explanatory variables, this at least can’t be applied immediately), assumptions of for example link function that are required to define coefficients, and especially with high-dimensional models, multicollinearity can become a notable problem.

In our model, we used specifically the logistic link function, implying $\sigma(x) = \frac{e^x}{e^x+1}$, although we acknowledge that other choices such as the probit link could have also been viable options. The estimation of the model was done with base r function glm. In order to curb issues related to multicollinearity, we first limited ourselves to the dataset with observations measured at multiples only observed at 16.8 meters. After this, we reduced the set of explanatory variables further via variance inflation factors.

According to the course material (James et al., 2021, p. 102), Variance inflation factors (VIFs) can be calculated for each explanatory variable in a model through assessing linear models for each of the explanatory variables. The VIF for explanatory variable X_j is defined as $\frac{1}{1-R_i^2}$ where R_i^2 is the amount of variance of X_j explained through a linear model by the other explanatory variables. Thus, these values express how much the explanatory variable in question can be described by a linear combination of the other explanatory variables. Thus, particularly high VIFs imply multicollinearity being present in the model. We calculated VIFs for our model with function vif from R package car.

As removing one explanatory variable effects the VIFs of all other variables, we chose to implement an iterative process to remove variables based on VIFs. After defining a logistic regression model with all explanatory variables, we calculated the VIFs, assessed which variable had the highest VIF, removed both this variable and the other variable describing the same set of observations (if a mean variable had the highest VIF, we would also remove the corresponding standard deviation and if a standard deviation variable had the highest VIF, we would also remove the corresponding mean variable). We decided to do this as it felt imprudent to choose to not include one descriptor of an observation in the baseline model which we’re assessing while including the other one. After this, we would repeat the process until the highest VIF value for a variable was less than 10. The choice of the limit was based on descriptions in the course material.

After this, we had a remaining set of 18 explanatory variables, which we modelled using logistic regression. After checking the coefficients, there were still variables for which the Wald tests gave relatively high p-values for hypotheses involving non-zero coefficients. In order to deal with this, we set those coefficients equal to zero and assessed the model with the remaining explanatory variables. Through this, we received a simpler model with explanatory variables.

Lasso regression

Lasso regression has obvious links to logistic regression but it includes a penalty term which punishes models with high values of coefficients. According to the course material (James et al., 2021, pp. 241–242) in comparable linear regression minimizing squared error, the minimizable function is the sum of the squared error terms and $\sum_{i=1}^{n} \gamma|\lambda_i|$ where $\gamma > 0$. We similarly as in logistic regression assess coefficients for a model that describes parameter q of a binomial model that minimizes this function.

One of the benefits of Lasso regression according to the High-Dimensional Statistics course material (Pirinen, 2021) is that including this penalty term in the optimization process with correlated variables leads to simpler models with fewer explanatory variables, as also lower-dimensional models lead are preferred. Thus, the lasso model in and of itself can remove variables which are highly correlated with other variables from the model and thus at least alleviate multicollinarity within the model. However, according to the High-Dimensional Statistics course material (Pirinen, 2021), selection between possible correlated variables might with extremely highly correlated variables break down due to there being multiple optima. Thus, it seems reasonable to note that with as highly correlated variables as the ones we're dealing with, the model might not properly describe the data-generating process but instead simply choose the variables that happen to create a slightly fit better on the training data.

In terms of our modelling, we decided to use Lasso instead of a comparable method called ridge regression, which according to the course material (James et al., 2021, pp. 237–238) uses penalty term $\sum_{j=1}^{m} \gamma\lambda_j^2$. The reason for this is that we preferred to make a simpler model with fewer explanatory variables rather than ridge regression, which tends to not reduce the dimension of the model but rather simply give similar coefficients to all correlated explanatory variables. In order to estimate the model, we used R functions glmnet and cv.glmnet from package glmnet (Friedman, Hastie, & Tibshirani, 2010). We attempted to assess the value of γ which would minimize loo cross-validated misclassification rate with the function cv.glmnet. After defining this value, we calculated cross-validated accuracy (which should be equivalent to what was calculated by cv.glmnet) as well as the cross-validated perplexity for the model estimated by glmnet.

Generative models

In logistic regression models, we directly model $p(Y|X)$, i.e. the conditional distribution of the response Y given the predictors X. Generative models are covered in (James et al., 2021, pp. 141–158). Generative models offer a less direct approach, where the focus is in modeling $p(X|Y)$ and using these estimates to estimate $p(Y|X)$ for each possible class k with Bayes' theorem:

$$p(Y = k|X) = \frac{\pi_k p(X|Y = k)}{\sum_i \pi_i p(X|Y = i)}.$$

Here pi_k is an estimate of the prior probability that a random observation comes from the kth class, in our case computed as the fraction of the observations in the training dataset that are in the kth class.

While the general approach is the same for all generative models, they differ in how they estimate $p(X|Y)$. In a dataset with p predictors, estimating $p(X|Y)$ amounts to estimating a p-dimensional density function for an observation in the kth class. The task is challenging, as we must consider the distribution of each predictor on its own *and* the joint distribution of the predictors. Different models make different assumptions that mitigate the difficulty.

Linear Discriminant Analysis A linear discriminant analysis (LDA) classifier assumes that all p predictors $X = (X_1, \ldots, X_p)$ are drawn from a multivariate Gaussian distribution. This means that each predictor follows a normal distribution $N(\mu_k, \Sigma)$ where μ_k is the class-specific mean and $\mathrm{Cov}(X) = \Sigma$ the covariance matrix of X, and there is some correlation between each pair of predictors. The estimate of $p(X|Y)$ is

$$p(X|Y) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma^{-1}(x - \mu_k)\right)$$

meaning that the classifier assigns an observation X to to the class for which

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

is maximized. The covariance matrix being the same for all classes is a defining feature of linear discriminant analysis. The assumption makes the model linear and reduces the number of parameters to estimate. This reduces the flexibility and hence lowers the variance of the model. Lower variance can mean that the model performs well, but if the assumption is not reflected in the data, bias can be high and overall performance suffers due to the bias-variance trade-off.

For linear discriminant analysis to work, it is important to have enough observations with regards to the number of predictors, as performance suffers greatly as the number of predictors approaches the number of observations.

In our modeling, we used the function lda from the package (Venables & Ripley, 2002). We tested the performance of the model on the dataset with observations measured at multiples only observed at 16.8 meters.

Quadratic Disriminant Analysis The assumption that the covariance matrix is shared for all classes is quite strict. Allowing each class to have it's own covariance matrix brings us to quadratic discriminant analysis (QDA). The observations are still assumed to be drawn from a multivariate Gaussian distrbution with a class-specific mean vector, so otherwise the assumptions remain the same as in LDA. Each predictor now follows a normal distribution $N(\mu_k, \Sigma_k)$. The expression to be maximized is

$$\delta_k(x) = x^T \Sigma_k^{-1} \mu_k - \frac{1}{2} x^T \Sigma_k^{-1} x - \frac{1}{2} \mu_k^T \Sigma_k^{-1} \mu_k - \frac{1}{2} \log |\Sigma_k| + \log \pi_k.$$

Generally, quadratic discriminant analysis is expected to perform better that linear discriminant analysis on large data sets, but as in linear discriminant analysis, the number of predictors must be small enough to produce decent results. Quadratic discriminant analysis is also recommended when the assumption that all classes share a covariance matrix is clearly unfounded.

In our approach, we used the qda function from the (Venables & Ripley, 2002) package, and as with linear discriminant analysis, tested the performance of the model on the dataset with observations measured at multiples only observed at 16.8 meters.

Naive Bayes A naive Bayes classifier makes no assumptions on the distribution of the observations, and instead assumes that within the kth class, the p predictors are independent. In other words, the assumption is that there is no association between the predictors, and therefore no joint distribution to consider. With this, the posterior probability can be computed as

$$p(Y = k|X) = \frac{\pi_k p(X_1|Y = k) \cdots p(X_p|Y = k)}{\sum_i \pi_i p(X_1|Y = i) \cdots p(X_p|Y = i)}.$$

In a naive Bayes model, there are a few possible ways to estimate $p(X|Y)$. For quantitative predictors such as ours, we assume that within each class, the jth predictor is drawn from a univariate normal distribution, i.e. $X_j|Y = k \approx N(\mu_{jk}, \sigma_j^2 k)$. The assumption of independence between predictors is often not realistic, but the model can perform well nonetheless. This is especially true for small datasets, since estimating a joint distribution requires a large amount of data. We used the naiveBayes function from the package (Meyer, Dimitriadou, Hornik, Weingessel, & Leisch, 2022) and the original dataset.

Support Vector Classifiers

The idea of the binary support vector classifier is according to the course material (James et al., 2021, pp. 369–370) based on the notion of finding within a p-dimensional problem a $(p - 1)$-dimensional hyperplane

$\lambda_0 + \sum \lambda_i x_i = 0$ to divide the set of explanatory variables. Now, for this hyperplane, we can divide our observations into two groups, those values for which $\lambda_0 + \sum_{i=1}^{m} \lambda_j x_i > 0$ and those values for which $\lambda_0 + \sum \lambda_i x_i < 0$. Thus, this can function as a model through which we can predict binary variables such as our variable of interest.

However, defining such hyperplanes can be slightly difficult as all values might not be separable by a hyperplane or the hyperplane that separates values can overfit to training data. Thus, according to the course material (James et al., 2021, pp. 373–375), we can define a soft-margin classifier, which has the benefits of letting some values be misclassified or be closer to the hyperplane than a strict maximal margin classifier would define, which leads to a more robust model that fits most of the training dataset better.

In order to define this, we need to find the optimal hyperplane to separate the observations. According to the course material (James et al., 2021, pp. 375–378), this optimization problem is an optimization problem of M in relation to coefficients λ_i and slack variables ϵ within inequation $y_i(\lambda_0 + \sum_{j=1}^{m} \lambda_j x_{ij}) \geq M(1 - \epsilon_i)$ with limitations $\sum_{j=1}^{m} \lambda_j^2 = 1$, $\epsilon_i \geq 0$ and $\sum_{i=1}^{n} \epsilon_i \leq C$ for tuning parameter C.

According to the course material (James et al., 2021, pp. 380–383), our optimization the data only depends on the inner products of the explanatory variables and the values of the variable of interest. Thus we can use different generalizations of inner products to gain different kinds of models. Because of this, with different choices of kernels we can model different kinds of data.

Other benefits benefits of using support vector classifiers according to the course material (James et al., 2021, pp. 367, 387) include that they have been proven to behave well in several different applications and that it has been shown that on average support vector classifiers perform better than logistic regression models in problems where the classes are well-separated. However, the model can vary quite notably based on the tuning parameter C as this parameter affects the amount of observations that violate the margin and thus increases or decreases the amount of support vectors that influence the model.

For our model, we used functions tune and svm from R package e1071 (Meyer et al., 2022). We chose to run a 10-fold cross validation on a set of tuning parameters C to assess which kernel performed best and chose to model the value based on that. In these preliminary assessments, we assessed the radial and linear kernels had the best performance of the ones available in package e1071. For these kernels, we ran loo-cross validation to assess the value of tuning parameter C and calculated for said tuning parameter loo-cross-validated accuracy and perplexity. Eventually, the radial kernel had slightly better accuracy but the linear kernel had lower perplexity and as a result, we chose to use the linear kernel in our model.

Classification trees

In tree-based methods, the data points are divided into smaller and smaller fragments based on some set of rules. A new division happens at each branch, so when the size of the tree grows, it becomes more and more fitted to the data. When fitting classification trees, the process is often first started with a very large tree, and then the unnecessary branches are removed in the process of pruning, until the complexity of the tree is reduces to the optimal level. Constructing the trees, pruning, and evaluating the results can be done in a variety of different ways.

A often used tree-based classification method is called Random Forest. It is based on the process described above, but only a random subset of the features is considered at each split of the tree. This is repeated multiple times and in the end, the obtained forest is evaluated and the best splits at each point are found. For Random Forest classifier, we used the R library *randomForest*.

Classifier performances (numerical)

In order to reduce knitting time for the file, we will in this report instead of expressing the full R code to calculate the values, simply report the results we received. The results for the two-class models in terms of loo-cross-validated accuracy and perplexity are:

Model type	Loo-CV accuracy	Loo-CV perplexity	Binary accuracy on test data
Dummy model	0	2.01	
Logistic regression	0.87	1.36	0.84
Naive Bayes	0.81	152.85	
LDA	0.89	1.33	
QDA	0.85	6.34	
SVM	0.88	-	0.84
Lasso	0.89	1.29	0.87
Random Forest	0.89	1.35	

Chosen classifier, pros and cons of this particular classifier for this application.

(Of note: the model is different from the one we originally used in the preliminary report and in the video, mostly because we noticed in our cross-validation calculations a reference error, which led to the loo-CV predictions being done on a dataset with including all observations, including the one being predicted, which naturally inflated our estimates of accuracy. When we noticed this error, we felt that no other models of the ones that had been observed were sufficiently close in terms of estimated perplexity so we chose to run Lasso with reduced amount of folds).

With our observed best-performing models being the the random forest model, the Lasso regression model and the linear discriminant analysis. Out of these models, we eventually chose to use the Lasso model as it was the best-performing model. The amount of information we could gain through CV calculations in R without R crashing were relatively limited, to the degree that we couldn't usually even calculate the cross-validated best choice for the tuning parameter. As a result, we chose to forego cross-validation in relation to our multinomial model.

We eventually also chose to not use the LDA model after assessing the assumptions related to the model. LDA models have been found to perform best with a limited amount of variables and we still question whether we should attempt to reduce the amount of variables from the model further from the current amount of 38 from the model. On top of this, considering the assumption of equal variances, the boxplots we drew earlier in the report imply that class "nonevent" might have larger variance for some variables than class "event".

We made the decision of model based on preliminary results, and though classification trees would represent an equally viable solution, we calculated those results only later on, and by that point had already calculated the results for the logistic regression model. However, for the final report we might consider transitioning to the classification tree model.

Thus we were left with the Lasso model. However, we could question whether for example specific variables need to be observed based on other research, and Lasso doesn't offer possibilities for this. Because cross-validation with a smaller amount of folds can give varying results different times it has been run, we eventually chose to run the cross-validation several times, check the interval of values received this way, and make decisions based on this interval.

Multiclass-classifier

We will extend the logistic Lasso model into a multinomial model in order to assess the multi-class accuracy for our model. According to the course material (James et al., 2021, pp. 140–141), the basic idea of multinomial logistic regression here is that out of K classes we choose a baseline value of our variable of interest, and compare other values to it, leading to $p(Y=k|X=x)=\frac{e^{\beta_{k0}+\sum_{j=1}^{m}\beta_{kj}x_{ij}}}{1+\sum_{k=1}^{K-1}e^{\beta_{k0}+\sum_{j=1}^{m}\beta_{kj}x_{ij}}}$ where our baseline class is class K. Alternatively, we can use equivalent so-called softmax coding where we treat all classes equivalently and thus $p(Y=k|X=x)=\frac{e^{\beta_{k0}+\sum_{j=1}^{m}\beta_{kj}x_{ij}}}{\sum_{k=1}^{K}-1e^{\beta_{k0}+\sum_{j=1}^{m}\beta_{kj}x_{ij}}}$. The interpretation of logistic regression coefficients comes now from the following: $\log(\frac{p(\hat{y}_i=k_a)}{p(\hat{y}_i=k_b)}) = (\lambda_{a0} - \lambda_{b0}) + \sum_{j=1}^{n}(\lambda_{aj} - \lambda_{bj})x_{ij}$. Thus the interpretation of these coefficients is in relation to the difference of the coefficients for different classes and describe the ratio between the probabilities that our variable of interest receives a value in one of these two classes.

Similarly as in the binary case, we attempt to define coefficients λ_j in a way that maximizes the pe.

In our model, we used the full dataset with no removed explanatory variables as we assumed the Lasso model itself would correct notable multicolinearity. In order to define this model, we used the same functions as with our binary model, cv.glmnet and glmnet from R package glmnet (Friedman et al., 2010). This creates estimated probabilities $\hat{p}(\hat{y}_i = y)$ for all four classes y, and through that we could calculate 10-fold cross-validated estimates for ideal values of the tuning parameter.

Eventually, we chose to emphasize class2 accuracy, and as a result our final predictions are based on estimating models for both class2 and class4. We first used our model for class2 to predict whether a value would be an event or not and for values the model assessing class2 predicted to be event-days, we referred to our class4 model to estimate which of the event classes had the highest probability.

Results

Although we defined our result comparisons with loo-CV which has the benefit of returning consistent results when run several times, according to the course material (James et al., 2021, pp. 205–206), for example 5-or 10-fold cross-validation can give more accurate assessments of error rate. Thus, for our binary model we ran 10-fold cross-validation 50 times, and received estimated ideal tuning parameters in interval [0.002073489, 0.04902743]. We first considered using the mean of these values as our choice of tuning parameter but as the values are rather dispersed (56% of values having tuning parameters $\lambda < 0.005$ but then again, we have approximately 30% of parameter values that are larger than 0.01), we should consider a different approach. Eventually, in order to bridge the gap between the two sets of values, we chose to prefer a slightly larger value than the median yet slightly lower than the mean for λ and thus chose value 0.0075 for the tuning parameter.

We eventually have a binary logistic regression model with explanatory variables which we assume describe the probability of a specific day being an event day. The coefficients of the model are:

There are a few things of note about this model. First, the intercept is fairly large, which implies that if all observations are at zero, the probability of an event would be very close to 1. The validity of this assumption should be studied further in relation to the relevant qualities of NPF events. Second, we notice that we have especially high absolute values of coefficients for the variables related to CS, although their somewhat high correlation (estimate 65%) and the small values received by both variables being rather small might explain this. Another thing we notice is that we have in particular variables related to the same phenomenon at different heights for the mean of carbon dioxide and the mean and standard deviation of RHIRGA. As we know these variables are highly correlated with each other, this ideal model might not be ideal. Then again, the coefficients are somewhat similar to each other and thus might simply describe an overall effect of the phenomenon (in a way reminiscent of optimization in Ridge regression).

For our multinomial model, we simply chose to observe the performance of the model on the data with different values of tuning parameter λ without cross-validation, as cross-validation even with a small number of folds couldn't be calculated with R for reasons that apparently might have to do with the small amount of data. Here we chose the value based on the multiclass accuracy of the model being sufficiently high with a non-negligible value of λ. We eventually chose a rather straightforward measure of choosing the largest value which led to over 50% of predictions being correct on the data on which it was being tested. The value was $\lambda = 0.004$.

In terms of accuracy, we couldn't calculate cross-validated predictions for our multinomial variable but let's observe results by class for our binomial model and see what types of events our model recognizes well.

Explanatory variable	Estimate of coefficient
Intercept	16.16886
CO2168.std	0.02065910
CO242.mean	-0.04024650
CO2504.mean	-0.02066992
H2O42.mean	-0.2627714
H2O84.std	-2.407117e-01
NO168.std	2.281875
NOx672.std	-0.06064781
O3672.mean	0.09233133
O384.std	0.03595632
Pamb0.std	0.2960355
PTG.mean	-15.19914
RGlob.mean	0.001578964
RHIRGA168.std	0.01765280
RHIRGA336.mean	-0.02661884
RHIRGA672.mean	-0.03528919
RHIRGA672.std	0.06932797
SO2168.std	0.7092859
SWS.mean	0.01112000
T168.std	0.4947142
CS.mean	-938.8368
CS.std	580.6009

Approximately 88% of true event days were predicted to be event days and similarly approximately 88% of true nonevent days were predicted to be nonevent days.

In terms of the binary model, we have rather similar accuracies for both event days and nonevent days. However, we also notice in terms of multiclass accuracy that while the model predicts in particular nonevents rather well, the LOO-cross-validated predictions are less successful for calculating distinct event types. For example, less than 10% of days which belonged to class Ia were predicted into said class. As a result, further changes should be made to the model in the future to improve these predictions.

Insights, conclusions, discussion etc.

In order to improve the model, we could ask several different questions concerning the approach we wound up with. First, in terms of variable selection, the measure of standard deviations per day within variables feels somewhat questionable in two different ways: first, it seems relevant to ask what the standard deviation expresses about the phenomena themselves, and whether different kinds of measurements could capture the relevant expressive ideas better, or whether these relevant expressive ideas even exist. Second, since we're assuming in our logistic regression model that the relationship is a linear combination or at least a linear combination of polynomials of the explanatory variables is relevant: why standard deviation and not variance? For this report, as we don't know sufficiently about the NPF event phenomenon itself, we will simply assume that the standard deviation variables are reasonable within our model and that this scaling is reasonable.

Second, in terms of our Lasso model, there are two big questions: first, our approaches towards correcting model complexity are rather limited, we can only really change it through changing our tuning parameter. Through changing our tuning parameter, we could get simpler or more complex models that might perform better on test data and we could in theory attempt to assess the value of the tuning parameter further. The second question relates to specific explanatory variables. If we know that we want to specifically include some specific variable with a non-zero coefficient in the model based on for example previous studies, we couldn't implement it within our current Lasso model whereas this would be possible in for example a logistic regression model. Thus we lose a certain kind of fleixibility in terms of variable selection, though we gain at least a model that functions reasonably well on the data we have.

Also, other variable selection processes for the same basic model (the logistic regression model) such as PCA could be used equivalently. However, for the current report, we chose to not use information criteria in relation to logistic regression models. We could have also used PCA but by using PCA, we'd lose the interpretability of the coefficients and when running the LOO-cv comparisons, we didn't receive better results in terms of accuracy or perplexity from using PCA compared to our current approach of selecting a sufficiently small number of explanatory variables through Lasso.

Self-grading

Delivarables

For the deliverables, we give a grade x.

For the most part, we are pleased with and proud of our project. We took great care in learning about the general theory of machine learning and different classifiers and their implementations. We prepared all our deliverables according to the instructions, and to the best of our ability. We specifically took the time to write a clear and high-quality report on our approach.

There are some specific issues that we're not completely satisfied with in retrospect. The first being the preprocessing of the data, and the second being the final accuracy and the details surrounding that.

A main contributor to the first problem was probably our work order: we wanted to preprocess the data, and then train and test the models. However, the course had not covered preprocessing while we were working on it, and although we consulted the course book liberally, our understanding and therefore the end result ended up being somewhat lackluster and disorganized.

For example, to reduce multicollinearity, we discarded most of the measurements at different heights because of the high correlation coefficients, and for many models this resulted in a higher accuracy and lower perplexity than what we achieved with the original dataset. Great! On the other hand, we left in the highly correlated radiation-related variables without much discussion or explanation. In short, we started with a preprocessing approach but did not see it to the end.

(Sara: muita preprocessing-probleemoja tai vastaavia, joita haluatte nostaa)

The second problem is perhaps not as grave as the first. Especially in our challenge submission, our accuracy (0.84 for two classes and 0.68 for four) and perplexity (1.47) were not all that great. This was partially due to a coding error which made us choose a model that perhaps wasn't ideal, but there was still room for improvement even with our other models. We agreed before submitting the results that we are not aiming to win anything, and the submission itself is enough. We stand by this. We still could have done better in predicting our own results, and we set to rectify this in our final report.

One of the other frustrations includes questions with our approach in relation to cross-validation. Perhaps the original choice of using loo-CV to assess the best model wasn't ideal and eventually in order to accomodate a lower amount of folds would have required us to possibly shift our entire approach from model accuracy and perplexity estimations to choice of ideal model and interpretation of said model as a result. Eventually we used a kind of messy half-measure where we decided our ideal model based on loo-CV and then for the model we used to predict the actual test data, we ran 10-fold cross validation to assess the ideal value of the tuning parameter. This was done to acknowledge the issues with the loo-CV even if, especially as we didn't enter into the contest with this model and for example the results in the table are still based on the loo-cv ideal model to compare it with the other models, and thus we didn't do much with the results we got.

Towards the 11th of December, we also found that many of our tree-based methods perform as well or better than logistic regression. If we had found this out earlier, we might have opted for a tree-based method as our final classifier. At that point we did not want to discard the work we had done so far, and opted to try to improve our logistic regression model instead.

Group as a whole

For the group, we give a grade 5.

We met very soon after being assigned to a group, and started planning and dividing work immediately. We agreed to use several different communication and project management tools, and while working settled on Telegram and weekly face-to-face meetings for communication, Excel for task management and storing key results, and Github for version control. We knew from the start that one member of the group would be absent during the week of the presentations, and planned accordingly.

In our group we had three majors represented: computer science, statistics and mathematics. We all had prior (but differing levels of) experience with R, and not much experience with machine learning. We were able to use our individual experiences to our advantage both in our discussions and while working on our R implementations and the report: we could clarify some of the more difficult concepts together, help each other with R troubleshooting, and read and comment on each other's work.

It was easy for us to trust each other's opinions and implementations, and trust that the agreed upon work gets done well and on time. At times, there were some lulls in the workflow and frustration because of R difficulties, but we were able to solve these situations quickly. We also felt similarly about our deliverables in the end: excellent in some ways, but lacking in others. We were able to have some very constructive discussions about this that highlighted the good atmosphere in the group, and how much we learned during the course to be able to spot the shortfalls in our approach.

Appendices

Appendix A

```
##        id             date              class4          partlybad      
##  Min.   :  1.0   Length:464         Length:464         Mode :logical  
##  1st Qu.:116.8   Class :character   Class :character   FALSE:464      
##  Median :232.5   Mode  :character   Mode  :character                  
##  Mean   :232.5                                                        
##  3rd Qu.:348.2                                                        
##  Max.   :464.0                                                        
##   CO2168.mean     CO2168.std        CO2336.mean      CO2336.std     
##  Min.   :359.6   Min.   : 0.05397   Min.   :359.1   Min.   : 0.04899  
##  1st Qu.:374.4   1st Qu.: 0.84564   1st Qu.:374.4   1st Qu.: 0.78959  
##  Median :380.8   Median : 1.95273   Median :380.7   Median : 1.89932  
##  Mean   :382.1   Mean   : 3.12997   Mean   :382.1   Mean   : 2.94065  
##  3rd Qu.:389.0   3rd Qu.: 4.42806   3rd Qu.:389.0   3rd Qu.: 4.14100  
##  Max.   :421.5   Max.   :19.46052   Max.   :421.1   Max.   :17.43986  
##    CO242.mean      CO242.std        CO2504.mean      CO2504.std     
##  Min.   :361.9   Min.   : 0.1115   Min.   :358.8   Min.   : 0.03742  
##  1st Qu.:375.4   1st Qu.: 0.9492   1st Qu.:374.3   1st Qu.: 0.78156  
##  Median :381.6   Median : 2.2723   Median :380.6   Median : 1.75885  
##  Mean   :383.0   Mean   : 3.9916   Mean   :382.0   Mean   : 2.71896  
##  3rd Qu.:389.7   3rd Qu.: 5.9603   3rd Qu.:389.0   3rd Qu.: 3.90350  
##  Max.   :422.6   Max.   :20.8517   Max.   :419.9   Max.   :16.65607  
##    Glob.mean         Glob.std        H2O168.mean       H2O168.std     
##  Min.   :  3.479   Min.   :  2.166   Min.   : 0.8702   Min.   :0.01335  
##  1st Qu.: 61.683   1st Qu.: 41.627   1st Qu.: 3.8563   1st Qu.:0.19028  
##  Median :194.669   Median :144.928   Median : 5.8639   Median :0.41783  
##  Mean   :188.905   Mean   :138.933   Mean   : 6.9032   Mean   :0.53393  
##  3rd Qu.:303.115   3rd Qu.:222.046   3rd Qu.: 9.4006   3rd Qu.:0.72848  
##  Max.   :426.457   Max.   :320.099   Max.   :18.7765   Max.   :2.87992  
##   H2O336.mean       H2O336.std      H2O42.mean       H2O42.std      
##  Min.   : 0.8872   Min.   :0.0110   Min.   : 0.8335   Min.   :0.02013  
##  1st Qu.: 3.8345   1st Qu.:0.1846   1st Qu.: 3.8911   1st Qu.:0.18787  
##  Median : 5.8037   Median :0.4169   Median : 6.0049   Median :0.41841  
##  Mean   : 6.8349   Mean   :0.5338   Mean   : 7.0239   Mean   :0.54499  
##  3rd Qu.: 9.3149   3rd Qu.:0.7380   3rd Qu.: 9.5254   3rd Qu.:0.73381  
##  Max.   :18.5736   Max.   :2.9379   Max.   :19.2872   Max.   :2.95080  
##   H2O504.mean       H2O504.std       H2O672.mean       H2O672.std     
##  Min.   : 0.8918   Min.   :0.01125   Min.   : 0.9047   Min.   :0.02052  
##  1st Qu.: 3.8171   1st Qu.:0.17681   1st Qu.: 3.7728   1st Qu.:0.17232  
##  Median : 5.7705   Median :0.41157   Median : 5.7575   Median :0.40763  
##  Mean   : 6.7958   Mean   :0.53064   Mean   : 6.7678   Mean   :0.53082  
##  3rd Qu.: 9.2741   3rd Qu.:0.73216   3rd Qu.: 9.2402   3rd Qu.:0.71825  
##  Max.   :18.4540   Max.   :2.97439   Max.   :18.4026   Max.   :3.05405  
##    H2O84.mean      H2O84.std           NET.mean         NET.std       
##  Min.   : 0.844   Min.   :0.02052   Min.   :-52.25   Min.   :  1.763  
##  1st Qu.: 3.870   1st Qu.:0.18912   1st Qu.: 37.61   1st Qu.: 37.932  
##  Median : 5.982   Median :0.41437   Median :117.82   Median :127.196  
##  Mean   : 6.973   Mean   :0.54114   Mean   :117.85   Mean   :121.780  
##  3rd Qu.: 9.468   3rd Qu.:0.72392   3rd Qu.:191.00   3rd Qu.:195.107  
##  Max.   :18.982   Max.   :2.87668   Max.   :302.83   Max.   :262.742  
##    NO168.mean         NO168.std          NO336.mean          NO336.std     
##  Min.   :-0.01438   Min.   :0.02096   Min.   :-0.01172   Min.   :0.02362
##  1st Qu.: 0.02114   1st Qu.:0.05182   1st Qu.: 0.02257   1st Qu.:0.05051
##  Median : 0.04335   Median :0.06355   Median : 0.04455   Median :0.06491
##  Mean   : 0.08533   Mean   :0.09767   Mean   : 0.09048   Mean   :0.09471
##  3rd Qu.: 0.08710   3rd Qu.:0.09328   3rd Qu.: 0.09161   3rd Qu.:0.09812
##  Max.   : 2.31625   Max.   :1.58946   Max.   : 2.39429   Max.   :0.86861
##    NO42.mean          NO42.std         NO504.mean         NO504.std
##  Min.   :-0.01222   Min.   :0.02218   Min.   :-0.02333   Min.   :0.02535
##  1st Qu.: 0.01890   1st Qu.:0.04952   1st Qu.: 0.02249   1st Qu.:0.05096
##  Median : 0.03717   Median :0.06286   Median : 0.04503   Median :0.06442
##  Mean   : 0.06939   Mean   :0.09971   Mean   : 0.08831   Mean   :0.09131
##  3rd Qu.: 0.07428   3rd Qu.:0.09086   3rd Qu.: 0.08847   3rd Qu.:0.09444
##  Max.   : 1.90179   Max.   :1.01254   Max.   : 2.23143   Max.   :0.88466
##    NO672.mean         NO672.std         NO84.mean           NO84.std
##  Min.   :-0.01327   Min.   :0.02310   Min.   :-0.02126   Min.   :0.02220
##  1st Qu.: 0.02320   1st Qu.:0.05053   1st Qu.: 0.01726   1st Qu.:0.04941
##  Median : 0.04474   Median :0.06500   Median : 0.03481   Median :0.06098
##  Mean   : 0.08599   Mean   :0.08921   Mean   : 0.07251   Mean   :0.09391
##  3rd Qu.: 0.08948   3rd Qu.:0.09387   3rd Qu.: 0.07372   3rd Qu.:0.08700
##  Max.   : 1.91714   Max.   :0.78343   Max.   : 2.06927   Max.   :1.37139
##   NOx168.mean       NOx168.std       NOx336.mean       NOx336.std
##  Min.   : 0.0949   Min.   :0.03606   Min.   : 0.0950   Min.   :0.0578
##  1st Qu.: 0.5082   1st Qu.:0.21941   1st Qu.: 0.5058   1st Qu.:0.2090
##  Median : 1.0234   Median :0.35092   Median : 1.0187   Median :0.3411
##  Mean   : 1.5248   Mean   :0.51310   Mean   : 1.5156   Mean   :0.5137
##  3rd Qu.: 2.0118   3rd Qu.:0.62159   3rd Qu.: 1.9926   3rd Qu.:0.6210
##  Max.   :12.6343   Max.   :6.26986   Max.   :12.5446   Max.   :6.0662
##    NOx42.mean         NOx42.std         NOx504.mean         NOx504.std
##  Min.   : 0.08883   Min.   : 0.06678   Min.   : 0.08358   Min.   :0.05679
##  1st Qu.: 0.53579   1st Qu.: 0.23237   1st Qu.: 0.49852   1st Qu.:0.20519
##  Median : 1.03089   Median : 0.36963   Median : 1.02123   Median :0.34535
##  Mean   : 1.53990   Mean   : 0.62994   Mean   : 1.50026   Mean   :0.52460
##  3rd Qu.: 1.97545   3rd Qu.: 0.70295   3rd Qu.: 1.95752   3rd Qu.:0.62162
##  Max.   :12.33232   Max.   :12.42314   Max.   :12.24750   Max.   :5.93928
##   NOx672.mean       NOx672.std       NOx84.mean        NOx84.std
##  Min.   : 0.0838   Min.   :0.05122   Min.   : 0.1005   Min.   :0.05788
##  1st Qu.: 0.5002   1st Qu.:0.20901   1st Qu.: 0.5051   1st Qu.:0.22016
##  Median : 1.0066   Median :0.34422   Median : 1.0289   Median :0.35778
##  Mean   : 1.4855   Mean   :0.50167   Mean   : 1.5231   Mean   :0.52454
##  3rd Qu.: 1.9644   3rd Qu.:0.59311   3rd Qu.: 2.0045   3rd Qu.:0.65374
##  Max.   :12.0379   Max.   :5.92112   Max.   :12.4547   Max.   :6.21296
##    O3168.mean       O3168.std         O342.mean         O342.std
##  Min.   : 3.613   Min.   : 0.2163   Min.   : 3.465   Min.   : 0.2001
##  1st Qu.:26.161   1st Qu.: 1.6010   1st Qu.:25.013   1st Qu.: 1.8771
##  Median :32.507   Median : 3.1526   Median :31.560   Median : 3.5260
##  Mean   :32.984   Mean   : 3.6238   Mean   :31.887   Mean   : 4.0472
##  3rd Qu.:39.931   3rd Qu.: 5.1053   3rd Qu.:39.189   3rd Qu.: 5.7574
##  Max.   :65.130   Max.   :13.5534   Max.   :63.968   Max.   :13.7123
##    O3504.mean       O3504.std         O3672.mean       O3672.std
##  Min.   : 3.834   Min.   : 0.2116   Min.   : 3.848   Min.   : 0.2111
##  1st Qu.:27.733   1st Qu.: 1.5723   1st Qu.:28.178   1st Qu.: 1.5831
##  Median :33.357   Median : 2.9209   Median :33.793   Median : 2.9202
##  Mean   :33.857   Mean   : 3.3877   Mean   :34.178   Mean   : 3.3024
##  3rd Qu.:40.583   3rd Qu.: 4.7962   3rd Qu.:40.783   3rd Qu.: 4.3776
##  Max.   :65.777   Max.   :12.8208   Max.   :65.775   Max.   :11.8905
##    O384.mean         O384.std          Pamb0.mean       Pamb0.std
##  Min.   : 3.603   Min.   : 0.09849   Min.   : 952.8   Min.   :0.06661
##  1st Qu.:25.564   1st Qu.: 1.68627   1st Qu.: 983.9   1st Qu.:0.45055
##  Median :31.991   Median : 3.27771   Median : 991.6   Median :0.79566
##  Mean   :32.394   Mean   : 3.78074   Mean   : 991.2   Mean   :1.02577
##  3rd Qu.:39.562   3rd Qu.: 5.42626   3rd Qu.: 997.9   3rd Qu.:1.28511
##  Max.   :64.505   Max.   :13.73501   Max.   :1018.8   Max.   :5.52491
##     PAR.mean          PAR.std           PTG.mean            PTG.std
##  Min.   :  6.258   Min.   :  4.764   Min.   :-0.0116443   Min.   :0.000000
##  1st Qu.:130.532   1st Qu.: 86.074   1st Qu.:-0.0028799   1st Qu.:0.004029
##  Median :387.052   Median :291.060   Median :-0.0002267   Median :0.008191
##  Mean   :373.349   Mean   :275.098   Mean   : 0.0001971   Mean   :0.009066
##  3rd Qu.:590.161   3rd Qu.:443.805   3rd Qu.: 0.0009356   3rd Qu.:0.013126
##  Max.   :825.394   Max.   :613.025   Max.   : 0.0910256   Max.   :0.040239
##    RGlob.mean       RGlob.std       RHIRGA168.mean   RHIRGA168.std
##  Min.   :-1.122   Min.   : 0.3671   Min.   : 27.71   Min.   : 0.196
##  1st Qu.:12.910   1st Qu.: 9.3438   1st Qu.: 54.50   1st Qu.: 2.520
##  Median :28.933   Median :21.3691   Median : 69.75   Median : 9.022
##  Mean   :27.834   Mean   :18.5654   Mean   : 69.49   Mean   : 8.480
##  3rd Qu.:42.247   3rd Qu.:27.0129   3rd Qu.: 88.22   3rd Qu.:12.430
##  Max.   :85.620   Max.   :49.7793   Max.   :106.02   Max.   :23.957
##  RHIRGA336.mean   RHIRGA336.std     RHIRGA42.mean     RHIRGA42.std
##  Min.   : 27.63   Min.   : 0.1972   Min.   : 29.66   Min.   : 0.2724
##  1st Qu.: 53.62   1st Qu.: 2.7418   1st Qu.: 54.98   1st Qu.: 2.1392
##  Median : 69.41   Median : 8.7971   Median : 69.73   Median : 9.7053
##  Mean   : 69.90   Mean   : 8.3927   Mean   : 70.12   Mean   : 8.8841
##  3rd Qu.: 88.82   3rd Qu.:12.3141   3rd Qu.: 87.99   3rd Qu.:13.5341
##  Max.   :107.44   Max.   :24.1008   Max.   :103.13   Max.   :23.5248
##  RHIRGA504.mean   RHIRGA504.std     RHIRGA672.mean   RHIRGA672.std
##  Min.   : 26.99   Min.   : 0.2573   Min.   : 26.70   Min.   : 0.1943
##  1st Qu.: 53.65   1st Qu.: 2.9149   1st Qu.: 54.56   1st Qu.: 2.9304
##  Median : 69.48   Median : 8.5568   Median : 70.02   Median : 8.2752
##  Mean   : 69.98   Mean   : 8.2231   Mean   : 70.70   Mean   : 8.1227
##  3rd Qu.: 89.07   3rd Qu.:12.1196   3rd Qu.: 90.03   3rd Qu.:12.0411
##  Max.   :105.74   Max.   :23.9752   Max.   :104.64   Max.   :24.0755
##  RHIRGA84.mean     RHIRGA84.std       RPAR.mean         RPAR.std
##  Min.   : 28.46   Min.   : 0.2591   Min.   :  0.00   Min.   : 0.000
##  1st Qu.: 54.38   1st Qu.: 2.3354   1st Qu.:  9.45   1st Qu.: 7.632
##  Median : 69.93   Median : 9.5532   Median : 18.87   Median :14.327
##  Mean   : 69.65   Mean   : 8.7783   Mean   : 20.16   Mean   :14.093
##  3rd Qu.: 87.93   3rd Qu.:13.1020   3rd Qu.: 26.07   3rd Qu.:17.487
##  Max.   :103.81   Max.   :23.6684   Max.   :134.86   Max.   :84.203
##   SO2168.mean        SO2168.std         SWS.mean         SWS.std
##  Min.   :-0.02743   Min.   :0.02887   Min.   :528.1   Min.   :  0.0000
##  1st Qu.: 0.05800   1st Qu.:0.07987   1st Qu.:907.4   1st Qu.:  0.7788
##  Median : 0.12286   Median :0.10977   Median :918.5   Median :  1.8926
##  Mean   : 0.27840   Mean   :0.17068   Mean   :908.2   Mean   : 20.0663
##  3rd Qu.: 0.30154   3rd Qu.:0.18339   3rd Qu.:923.1   3rd Qu.: 16.8917
##  Max.   : 3.81492   Max.   :2.01559   Max.   :937.9   Max.   :190.6516
##    T168.mean         T168.std          T42.mean           T42.std
##  Min.   :-24.778   Min.   :0.04589   Min.   :-24.883   Min.   :0.05069
##  1st Qu.: -1.530   1st Qu.:0.72766   1st Qu.: -1.510   1st Qu.:0.75975
##  Median :  6.402   Median :1.80772   Median :  6.621   Median :1.98491
##  Mean   :  5.681   Mean   :1.79699   Mean   :  5.752   Mean   :1.95714
##  3rd Qu.: 13.720   3rd Qu.:2.68334   3rd Qu.: 13.800   3rd Qu.:2.92745
##  Max.   : 27.719   Max.   :5.14434   Max.   : 27.889   Max.   :5.09896
##    T504.mean          T504.std         T672.mean          T672.std
##  Min.   :-24.017   Min.   :0.05055   Min.   :-23.901   Min.   :0.05258
##  1st Qu.: -1.804   1st Qu.:0.69842   1st Qu.: -1.991   1st Qu.:0.68697
##  Median :  6.118   Median :1.71758   Median :  5.959   Median :1.65568
##  Mean   :  5.384   Mean   :1.65693   Mean   :  5.189   Mean   :1.59360
##  3rd Qu.: 13.383   3rd Qu.:2.50350   3rd Qu.: 13.040   3rd Qu.:2.35899
##  Max.   : 27.276   Max.   :5.06112   Max.   : 27.110   Max.   :4.86725
##     T84.mean          T84.std       UV_A.mean            UV_A.std
##  Min.   :-24.875   Min.   :0.0439   Min.   : 0.2959   Min.   : 0.1778
##  1st Qu.: -1.477   1st Qu.:0.7615   1st Qu.: 4.2367   1st Qu.: 2.4317
##  Median :  6.577   Median :1.9454   Median :11.3274   Median : 7.5885
##  Mean   :  5.767   Mean   :1.9113   Mean   :10.6230   Mean   : 7.4510
##  3rd Qu.: 13.832   3rd Qu.:2.8284   3rd Qu.:16.4605   3rd Qu.:11.8278
##  Max.   : 27.939   Max.   :5.1320   Max.   :22.5412   Max.   :16.8305
##    UV_B.mean          UV_B.std           CS.mean              CS.std
##  Min.   :0.00514   Min.   :0.003552   Min.   :0.0002433   Min.   :2.727e-05
##  1st Qu.:0.12586   1st Qu.:0.086265   1st Qu.:0.0013907   1st Qu.:2.661e-04
##  Median :0.40225   Median :0.334264   Median :0.0023979   Median :4.759e-04
##  Mean   :0.42880   Mean   :0.366484   Mean   :0.0029625   Mean   :6.673e-04
##  3rd Qu.:0.66997   3rd Qu.:0.589098   3rd Qu.:0.0039100   3rd Qu.:7.908e-04
##  Max.   :1.19727   Max.   :1.055615   Max.   :0.0126701   Max.   :6.277e-03
##      class2
##  Min.   :0.0
##  1st Qu.:0.0
##  Median :0.5
##  Mean   :0.5
##  3rd Qu.:1.0
##  Max.   :1.0
```

Sources

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, *33*(1), 1–22. doi:10.18637/jss.v033.i01

Hornik, K., & Böhm, W. (2022). *Clue: Cluster ensembles.* Retrieved from https://cran.r-project.org/package=clue

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning with Applications in R* (Second edition.). Springer.

Maso, M. D., Kulmala, M., Riipinen, I., Wagner, R., Hussein, T., Aalto, P. P., & Lehtinen, K. E. J. (2005). Formation and growth of fresh atmospheric aerosols: Eightyears of aerosol size distribution data from SMEAR II, Hyytiälä, Finland. *BOREAL ENVIRONMENT RESEARCH.* Retrieved from http://www.borenv.net/BER/archive/pdfs/ber10/ber10-323.pdf

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2022). *e1071: Misc functions of the department of statistics, probability theory group (formerly: E1071), TU wien.* Retrieved from https://CRAN.R-project.org/package=e1071

Ng, A., & Jordan, M. (2001). On discriminative vs. Generative classifiers: A comparison of logistic regression and naive bayes. In T. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems* (Vol. 14). MIT Press. Retrieved from https://proceedings.neurips.cc/paper/2001/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf

Pirinen, M. (2021). *HDS 6. Penalized regression.* Retrieved from https://www.mv.helsinki.fi/home/mjxpirin/HDS_course/material/HDS6_penalized_regression.html

R Development Core Team. (2008). *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from http://www.R-project.org

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with s* (Fourth.). New York: Springer. Retrieved from https://www.stats.ox.ac.uk/pub/MASS4/

Wei, T., & Simko, V. (2021). *R package 'corrplot': Visualization of a correlation matrix.* Retrieved from https://github.com/taiyun/corrplot