Term Project - Group 80

Sari Ropponen, Outi Boman, Loic Dreano

2022-12-21

Description of data

In the project a data set of new particle formation, NPF, is used. The data is divided into training data (npf_train.csv) and testing data (npf_test_hidden.csv) and it include 104 variables relating to daily mean and standard deviation measurements taken in Hyytiälä forestry field station. The number of observations in training data is 464 and 965 in testing data.

Table 1: Summary of the dataset

	npf_train	npf_test
Measurements	464	965
Variables	104	104

Some of the variables like temperature T and carbon dioxide concentration CO2 are measured in different heights. The height is indicated in the name of the variable, for example, variable "T84.mean" is the mean temperature at 8.4 meters above the mast base during a day. The data includes also a response variable "class4" indicating if a NPF event has happened during the day or not. In the project a binary classifier is build to predict if an NPF event happens or not during a day according to observed measurements.

Preprocessing data

The data includes variables that are not used as predictors. A summary of the the variables in training data is below:

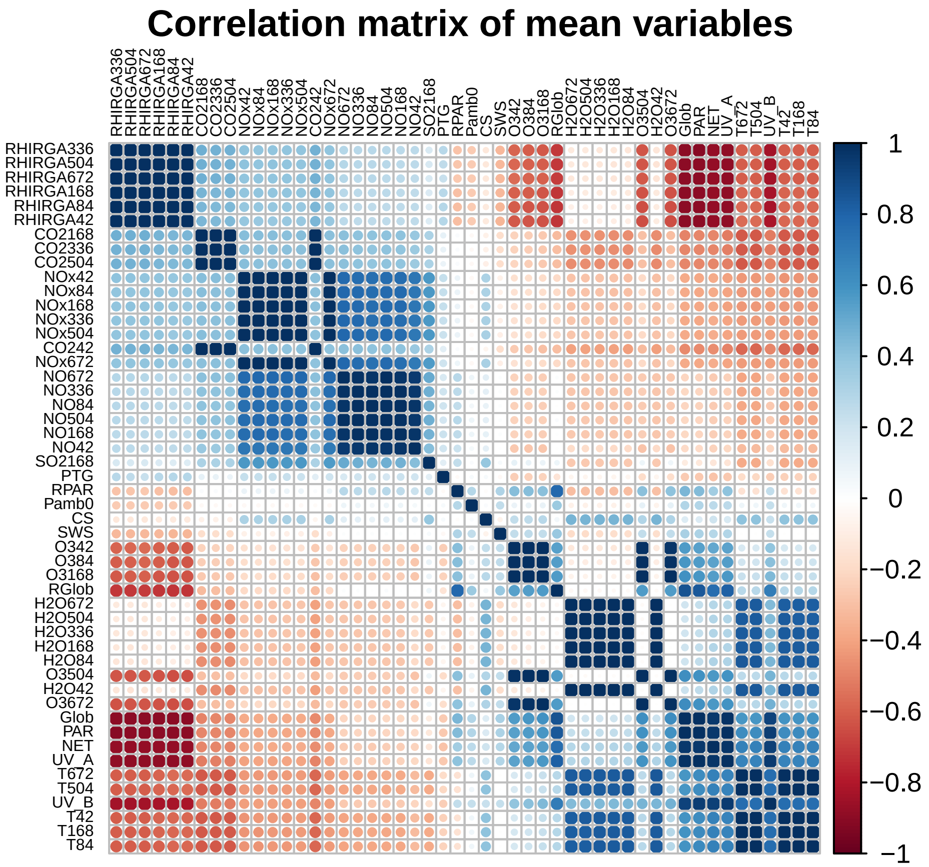
```
##        id             date            partlybad
##  Min.   :  1.0   Length:464         Mode :logical
##  1st Qu.:116.8   Class :character   FALSE:464
##  Median :232.5   Mode  :character
##  Mean   :232.5
##  3rd Qu.:348.2
##  Max.   :464.0
```

The column "date" was set to be the row names in the training data. Because the value of the logical variable "partlybad" is FALSE for all the observations, it doesn't give any information. Columns "id", "date" and "partlybad" were removed from the data.

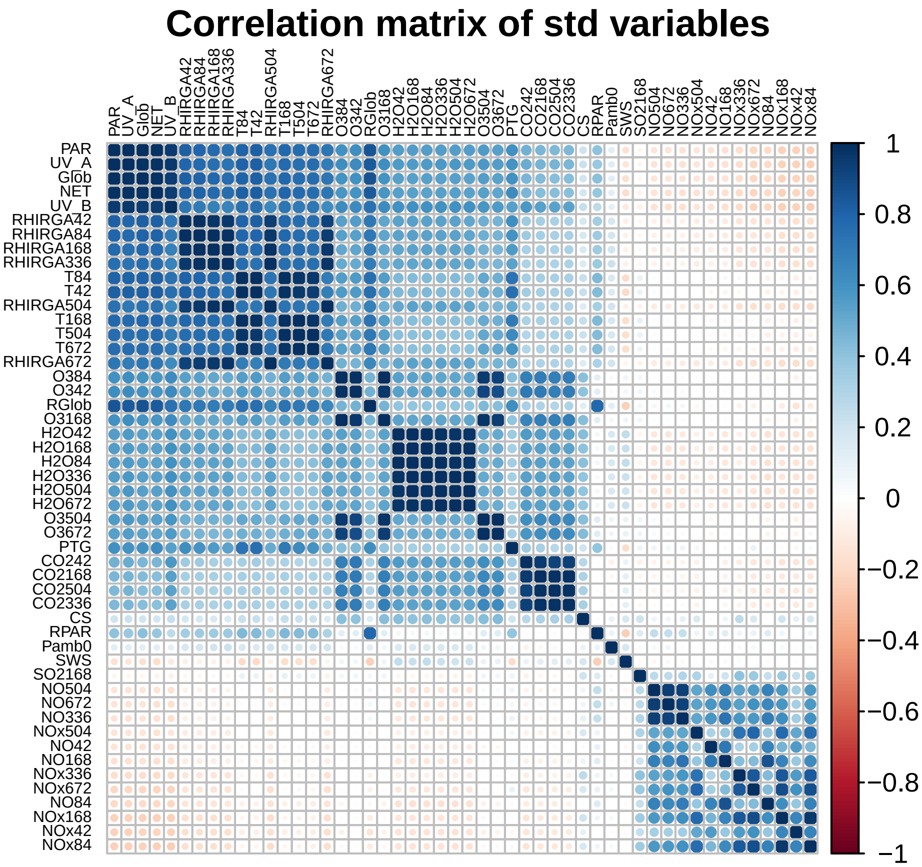
A qualitative variable "class2" is added to the training data. It gets either value "event" or "nonevent" according to "class4". Variable "class4" indicates the type of the event if it has happened ("Ia", "Ib" or "II") or "nonevent" if no event has happened during the day.

Exploration of the training data

Because the data includes the same measurements at different heights it is expected that there are correlation between variables. Correlations between different mean values are shown in the matrix below:



The correlation matrix for standard deviations is



As we can see, there are a lot of highly correlated variables and since they carry the same information, keeping all of them would not improve the predictivity of our models. In order to increase the power of our models to identify independent variables that are statistically significant and to make them simpler to interpret (simpler model in general) we remove the highly correlated variables.

To do so the variables are clustered together based on their correlation; every pairs of variables which have an absolute correlation greater than 0.8 are clustered together. Then a random variable from each cluster is kept for further analysis.

The correlations for mean and standard deviations after cleaning the data are shown in the pictures below as well as a table of variables left in the data (table 2). Table 3 shows that the data includes only 26 variables. Histograms of the variables left in the data are in the annex. The histograms give us an idea of the values of the variables. The information is used, for example, in PCA to decide whether to normalize and scale the data or not.

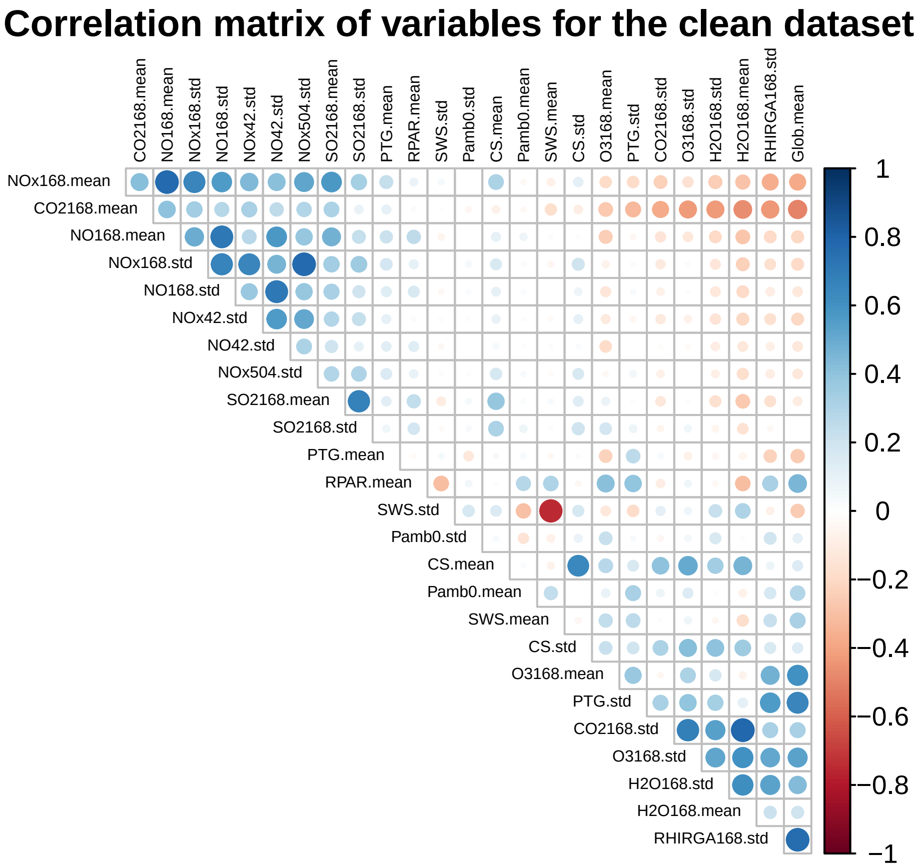


Table 2: List of kept variables

NO42.std	SO2168.mean	CO2168.std	O3168.mean
NOx42.std	SO2168.std	H2O168.mean	O3168.std
NOx504.std	SWS.mean	H2O168.std	RHIRGA168.std
Pamb0.mean	SWS.std	NO168.mean	RPAR.mean
Pamb0.std	CS.mean	NO168.std	Glob.mean
PTG.mean	CS.std	NOx168.mean	NA
PTG.std	CO2168.mean	NOx168.std	

Table 3: Summary of clean dataset

	npf_train	npf_test
Measurements	464	965
Variables	26	26

Performance measures

To compare different classifiers two measures are used: accuracy and perplexity. Accuracy is the proportion of the observations that has been classified correctly. Perplexity is a rescaled variant of log-likelihood. If perplexity equals to 1 the classifier predicts always the probability of an observation to an actual class. Perplexity of 2 corresponds to coin flipping.

If the method predicts a probability of an event and the estimated posterior probability is more than 0.5, observation is classified as “event”. Otherwise the observation is classified as “nonevent”.

Performance measures are calculated using validation method and 10-fold Cross-Validation on the training data. For validation method the training data is randomly divided into 2 equally sized data sets: training data to fit the model and validation data to estimate the accuracy and perplexity. Accuracy and perplexity are also calculated in training data for comparison and to evaluate if there is overfitting.

Generalization accuracy and perplexity is estimated using cross-validation to the training data. The data is randomly divided into 10 folds and performance measures are calculated using each fold as a validation data in turn. A single run of the k-fold cross-validation procedure may result in a noisy estimate of model performance. In fact, different splits of the data may result in very different results. Thus, the cross-validation procedure is repeated 100 times and the mean result across all folds from all runs are reported.

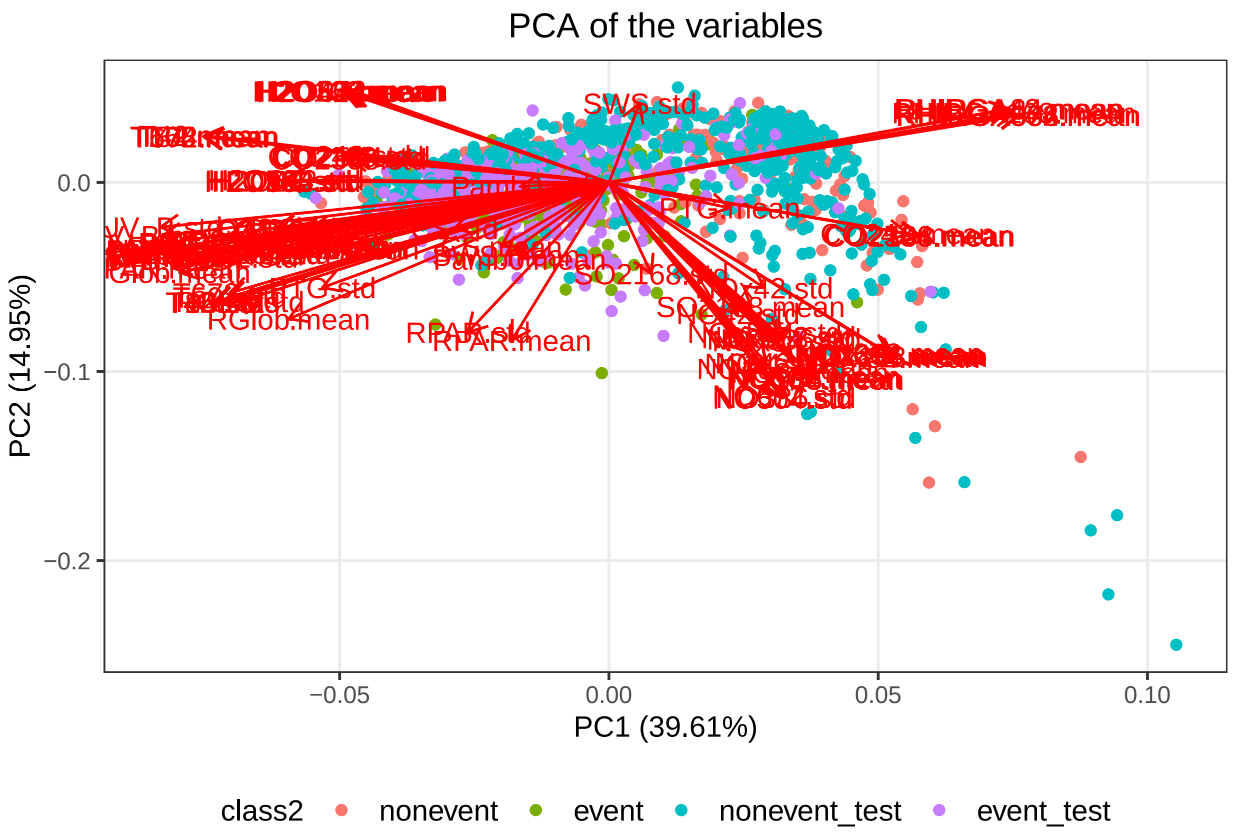
The increase of the repetition increases also the risk to get values of 1 or 0 for probabilities, and in that case the calculation of the perplexity isn't possible. Thus, in order to get a finite number result, the values 0.000000001 and 0.999999999 are used instead of 0 and 1 respectively.

Investigation of the variables: PCA

Principal Component Analysis (PCA) is used to study how much we can still reduce dimensionality of the data but to save as much of the variability (that is, information) of the data as possible at the same time. As we can see from the histograms of the variables (see the annex), the variables are measured in different units with different magnitudes of variances. Thus, the variables are centered to have zero mean and scaled to have standard deviation one.

We do the PCA using the original training data (npf_train.csv) combined with the original testing data (npf_hidden.csv).

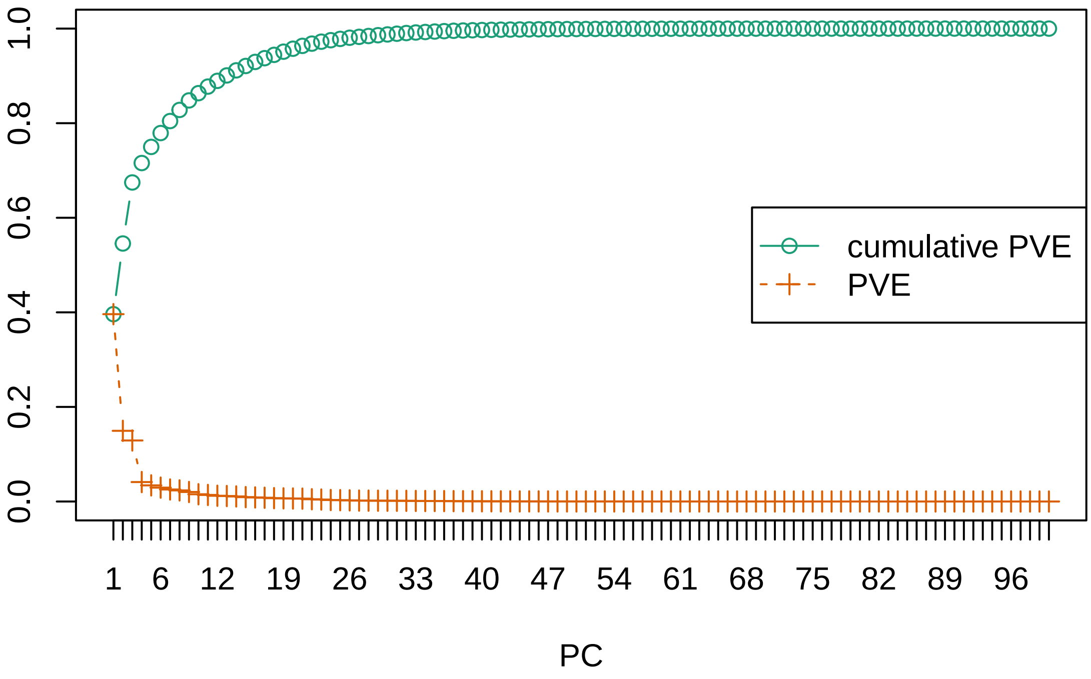
First, we do PCA before cleaning the correlated variables. The biplot of the analysis is below:



The biplot includes the first two principal components (PCs), both the scores and loading vectors. Also, the proportion of variance explained (PVE) by the PC is indicated. As we can see, many of the vectors are overlapping meaning that they are correlated. This gives us a further justification to remove the correlated variables and leaving only one of each correlation cluster in the data. We can also see that the first PC (PC1) explains only 39.61% of the variance in the data and the second PC (PC2) $14,95\%$. Together they explain only $54,56\%$ of the variance which is quite poor result the target being more than 80%.

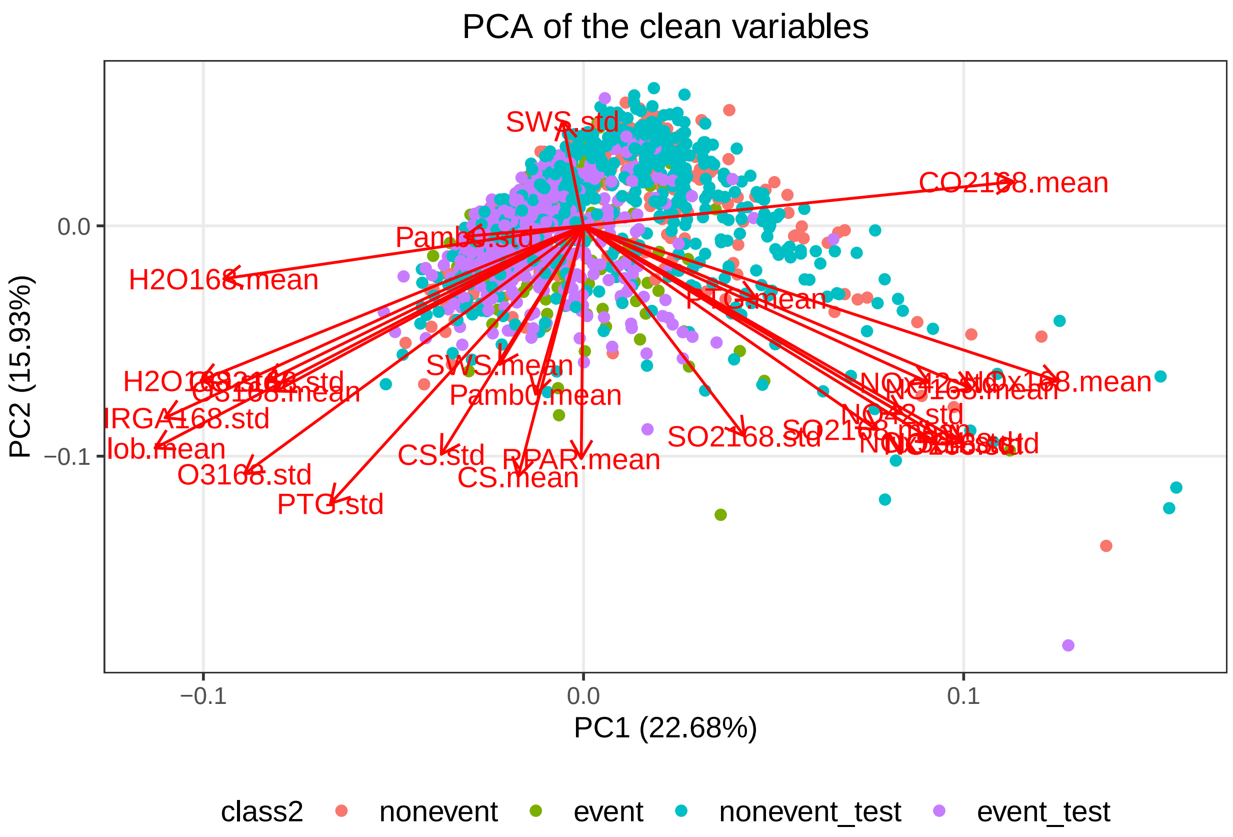
From the biplot we can also see that there are a couple of outliers in the bottom right corner. For now, we leave them to be and handle them during modelling if needed.

To see how many PCs are needed to explain more than 80% of the variance the PVE by each principal component and the cumulative PVE is shown in the figures below:



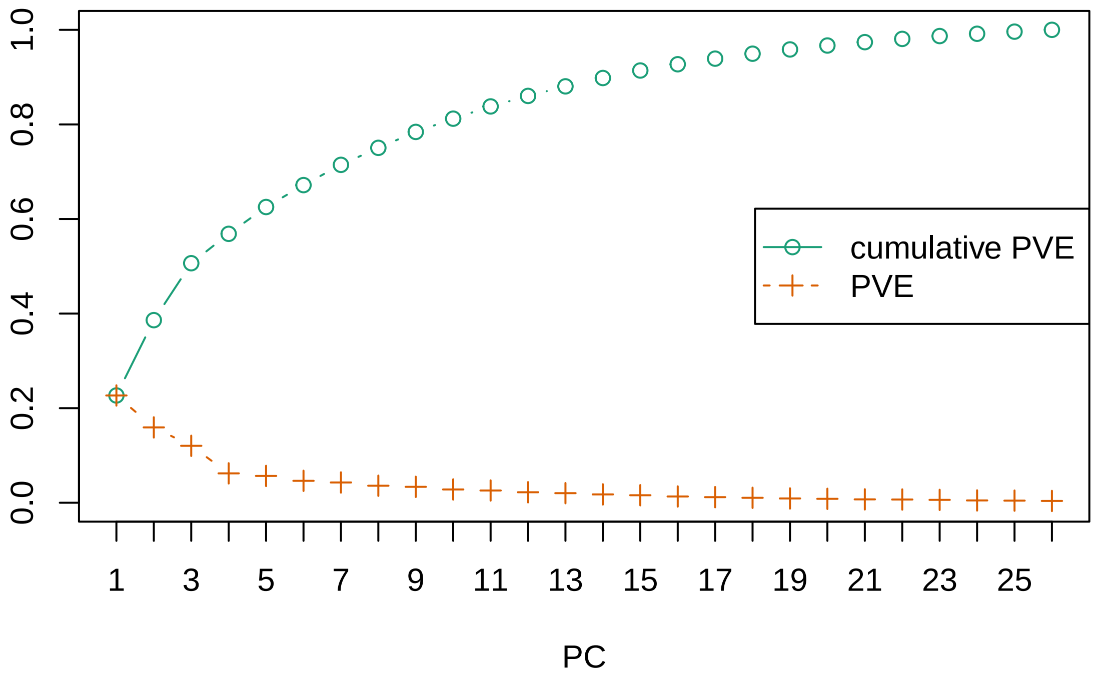
For the original data we would need around $5 - 10$ PCs to explain most of the variance.

To see how the results are changed after cleaning the data from correlated variables PCA is also done for the cleaned data. The biplot is



The loading vectors are not as overlapped as previously but the PVE is still quite poor: the two first PCs explains only $38,61\%$ of the variance.

Again, PVE by each principal component and the cumulative PVE are investigated:



We can see that about 10 first principal components should be used to explain about 80% of the variance.

The PCA shows that we could still reduce the dimensionality of the data from 26 variables to some extend. We are going to do that later by using the results of the PCA together with Naive Bayes. In next section we first investigate some other methods which already include feature selection.

Models including feature selection

Some methods include variable selection in itself, like Lasso and decision trees. In the project We investigate logistic regression with Lasso and random forest.

1) Logistic regression with Lasso

Logistic regression is a discriminant classifier which assumes that the log odds is linear in variables. When combined with Lasso a subset selection of variables are done by adding a so called penalty term to residual sum of squares which is minimized in parameter estimation process. The bigger the estimated coefficients of the variables are the bigger the penalty. The penalty term forces some of the coefficients to zero resulting in variable selection at the same time. As we can see from the histograms in the annex the variables are measured in different units and scales. To make sure that the magnitude of a variable does not give too much weight we use data that is normalized to zero mean and scaled to unit variance.

The amount of penalty depends on a coefficient called lambda. The value of lambda is selected by cross-validation so that the value of the test error is minimized.

The estimated coefficients and the lambda used in training data (with 232 observations) are

term	step	estimate	lambda	dev.ratio
(Intercept)	1	-0.050	0.009	0.69

term	step	estimate	lambda	dev.ratio
NO42.std	1	0.009	0.009	0.69
Pamb0.std	1	0.199	0.009	0.69
PTG.std	1	0.263	0.009	0.69
SO2168.std	1	0.069	0.009	0.69
SWS.mean	1	0.582	0.009	0.69
CS.mean	1	-1.712	0.009	0.69
CS.std	1	0.267	0.009	0.69
CO2168.mean	1	-0.527	0.009	0.69
CO2168.std	1	0.250	0.009	0.69
H2O168.mean	1	-0.929	0.009	0.69
NO168.std	1	0.313	0.009	0.69
O3168.mean	1	1.443	0.009	0.69
O3168.std	1	0.230	0.009	0.69
RHIRGA168.std	1	0.379	0.009	0.69
Glob.mean	1	0.956	0.009	0.69

As we can see, a variable selection has been done since only 15 variables have a nonzero coefficient and some of them are still close to zero. The given dev.ratio 0.69 is unfortunately not very good.

2) Random Forest

Decision trees are learning methods that segment observations into regions. The segmentation is done recursively using binary decision rules which minimize the selected measure like residual sum of squares (RSS) in case of regression tree or e.g. classification error rate or Gini index in case of classification tree. A prediction for an observation belonging to a certain region is given by the mean of responses of the training observations in the same region (regression tree) or by the label to which the majority of the training observations belong to in the same region (classification tree).

Random Forest builds a number of decision trees using each time a sample of training data, sampling done by bootstrapping. Predictions are averages of the resulting trees. During each split in tree growing process only one of the given number of randomly selected variables are considered. By reducing the number of variables, which are consider during each split, the correlation between different trees is reduced.

To get predictions of probabilities of event given the observations we redefine the response variable class2 to be a dummy variable: it gets a value 1 when an event has occurred and value 0 in case of nonevent. Nine variables are considered during each split. The selected number equals to the commonly used one third of the variables in the data in case of regression tree. We use normalized and scaled data.

Confusion matrix for training data is

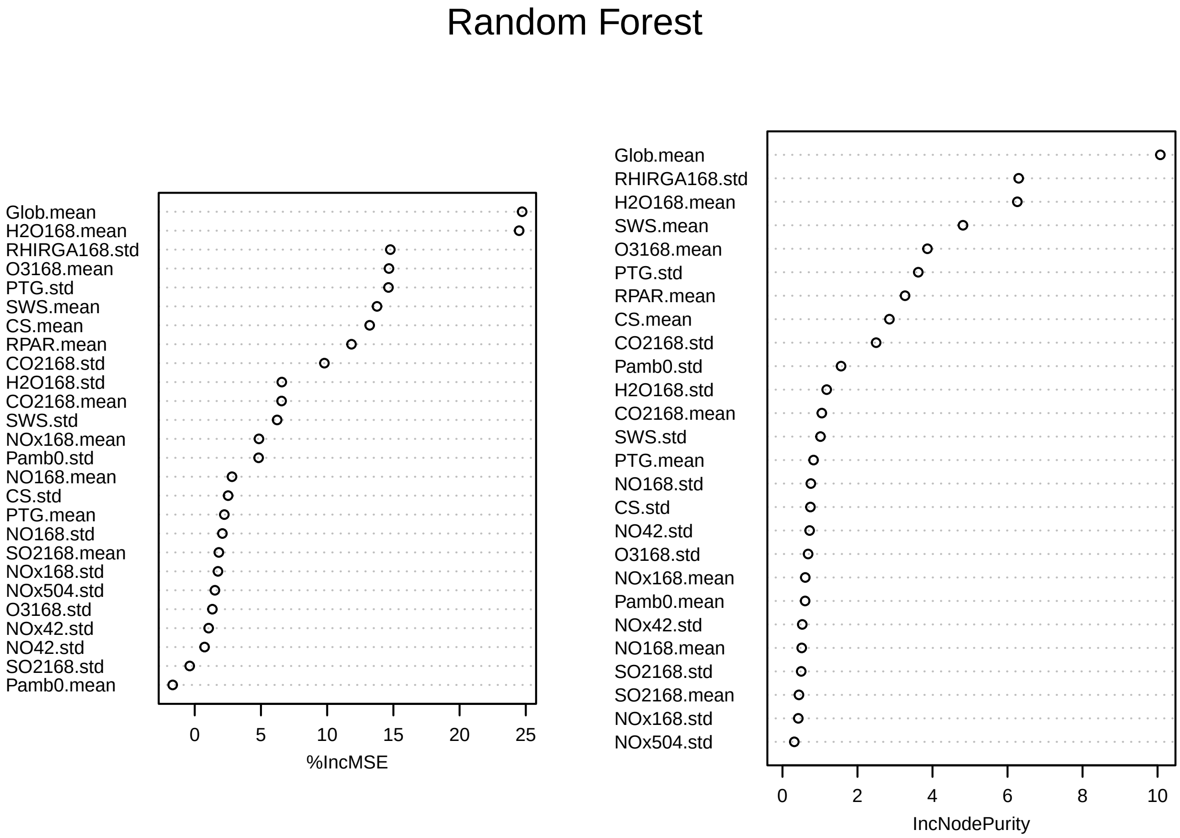
	nonevent	event
0	115	0
1	0	117

As we can see, no missclassification is done. To see how well the model performs on validation set, we look at the confusion matrix for validation data set:

	nonevent	event
0	102	20
1	15	95

There are 20 observations predicted to be nonevent even though event has occurred and 15 observations with the false prediction of event. All together, the accuracy in validation set is $84,9\%$.

The importance of variables are:



The picture above report the values of two variables: Mean Decrease Accuracy (%IncMSE) that shows how much our model accuracy decreases if we leave out that variable and Mean Decrease in MSE (IncNodePurity) that is a measure of variable importance based on the MSE.

The higher the value of mean decrease accuracy or mean decrease Gini score, the higher the importance of the variable to our model. We can see that random forest do some kind of a variable selection since only some of the variables have a significant importance in forming the regions and thus, the predictions. However, the most important variables are not the same as with logistic regression with Lasso.

Perfomance of the models

The performance of the classifiers investigated so far are:

Model	Train Accuracy	Validation Accuracy	CV Accuracy	Train Perplexity	Validation Perplexity	CV Perplexity
Log reg Lasso	0.918	0.828	0.873	1.24	1.343	1.337
Random Forest	1	0.849	0.497	1.107	1.389	4.191

Even though the accuracy of random forest on validation set is better than of logistic regression with Lasso the performance of random forest estimated by cross-validation is poor. All together, logistic regression with

Lasso clearly outperforms random forest. Especially, cross-validated perplexity of logistic regression with Lasso seems to be good.

Other models

The models tested in addition to the logistic regression with Lasso and random forest are a dummy model, Naive Bayes with reduced dimensionality, logistic regression using all variables in cleaned data (to compare the results with the one with Lasso) and k nearest neighbor.

1) Dummy

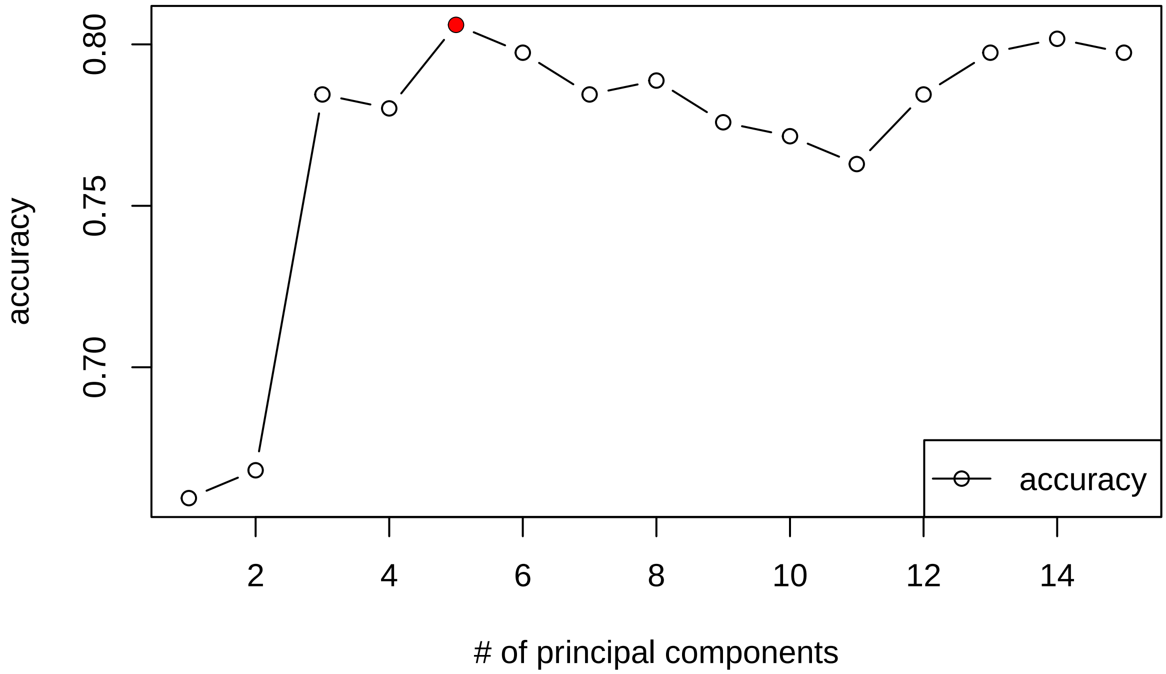
Dummy classifier makes predictions based only on classes or labels and it ignores all the input features of the data. Even though it is not highly performing classifier, it is still a valuable feature as it is usually used as a baseline to compare the performance of the other methods.

In this project the dummy classifier uses the most frequent class of the training set for the predictions. In another method the frequencies of the classes in a training data could be used as the probabilities for choosing between the classes. Some dummy classifiers does not even calculate the frequencies as they predict the classes randomly using a uniform distribution.

2) Naive Bayes

Naive Bayes is a generative classifier which assumes that the variables in each class are independent and are from Gaussian distribution. It can be assumed that the assumption of independence is not the case in NPF data, but Naive Bayes can still produce quite decent results. No scaling of the data is needed since it doesn't affect the predictions.

In addition to using Naive Bayes with all 26 variables in the cleaned data, we use Naive Bayes after reducing dimensionality by PCA. This means, that we use the principal component score vectors as an input to the model. To decide the number of the PCs to use, we look at the accuracy of the model in validation data with different number of first PCs used (see the picture below). The highest accuracy is marked with red.



The highest accuracy is received when using the five first PCs, so this model is added to the comparison of methods.

3) Logistic regression

To compare the results with logistic regression with Lasso we train also logistic regression on the training data with all the 26 variables. We use normalized and scaled data to avoid dominance of variables measured in high magnitudes.

4) k Nearest Neighbor (k-NN)

In k-NN an observation is labeled with the class which occurs most often within the k closest training data points to the observation (that is, the class with the largest proportion). Because the distance is used the data needs to be scaled. Parameter k controls the flexibility of the classifier. The smaller the k the more flexible the decision boundary is. With a very small k the classifier can classify the training data more accurately but this may end up to overfitting.

Different values of k are tried to choose the optimal k. Accuracy on the validation set for each k are shown in the table below:

k	accuracy
1	0.78
5	0.836
10	0.836
15	0.828
20	0.823
50	0.78

When k is 5 or 10, the accuracy is highest. To avoid overfitting we select k = 10 to be one of the methods to compare with other methods.

Comparison of the methods

Performance measures for the methods are:

Model	Train Accuracy	Validation Accuracy	CV Accuracy	Train Perplexity	Validation Perplexity	CV Perplexity
Log reg Lasso	0.918	0.828	0.873	1.24	1.343	1.337
Random Forest	1	0.849	0.497	1.107	1.389	4.191
Dummy	0.504	0.496	0.506	1.983	2.017	2
Naive Bayes	0.853	0.746	0.753	2.106	4.122	7.908
Naive Bayes with PCA	0.875	0.806	0.816	1.46	1.471	1.625
Logistic regression	0.94	0.862	0.886	1.184	1.439	1.365
10-NN	0.897	0.862	0.844	1.278	1.795	1.601

As we already concluded logistic regression with Lasso performs better than random forest. Logistic regression with all 26 variables performs surprisingly well. However, perplexity is better when variable selection is done by Lasso. Cross-validated performance of random forest is also poor compared to other methods. Only Naive Bayes' perplexity is worse than random forest.

Naive Bayes with all 26 variables performs poorly producing high perplexities. When the dimensionality is reduced by PCA the performance improves significantly but does not outperform logistic regression models. Perfomrance of 10-NN is close to the performance of Naive Bayes with PCA.

As expected, the performance of dummy model is not good and it corresponds guessing if an event happens or not (like coin flipping). However, Naive Bayes performs even worse than dummy model at least when we look at perplexities.

According to the investigated performance measures logistic regression with Lasso is selected as a classifier. It has the highest cross-validated accuracy and lowest cross-validated perplexity, and the results are at the same level on validation set, too.

Summary

We started by looking at the data and especially the correlations between variables. Correlated variables were removed from the data to improve the predictivity of the models. PCA showed that dimensionality could still be somewhat reduced. In stead of further selecting a subset of the 26 variables we used methods which included features of variable selection, like Lasso combined with logistic regression, random forest and PCs with Naive Bayes. When appropriate we used normalized and scaled data to avoid the variables with high magnitudes to dominate the model fitting and predictions.

According to the performance comparison we selected logistic regression with Lasso as our binary classifier. It is used on normalized and scaled data. We noticed that if the unscaled data is used the model will predict only events on testing data. The selected model gives the highest accuracy and the smallest perplexity in cross-validation and the performance is stable between validation approach and cross-validation. Pros of this method are that the variable selection is done by Lasso and the fact that logistic regression doesn't make any assumptions of the distribution of the observations in different classes. We also have enough data compared to the number of variables in cleaned data to train the model properly. By using Lasso to reduce number of variables we also reduce the possibility of overfitting the model. One of the disadvantages of the model is the assumption of linear decision surface which is not usually valid. However, the method outperformed for example random forest which can handle more complex relationships.

Performance of the selected method on testing data

Selected method was trained using the original, cleaned training data (npf_train.csv) and predictions on testing data (npf_hidden.csv) was generated. Unfortunately, the results turned out to be very poor:

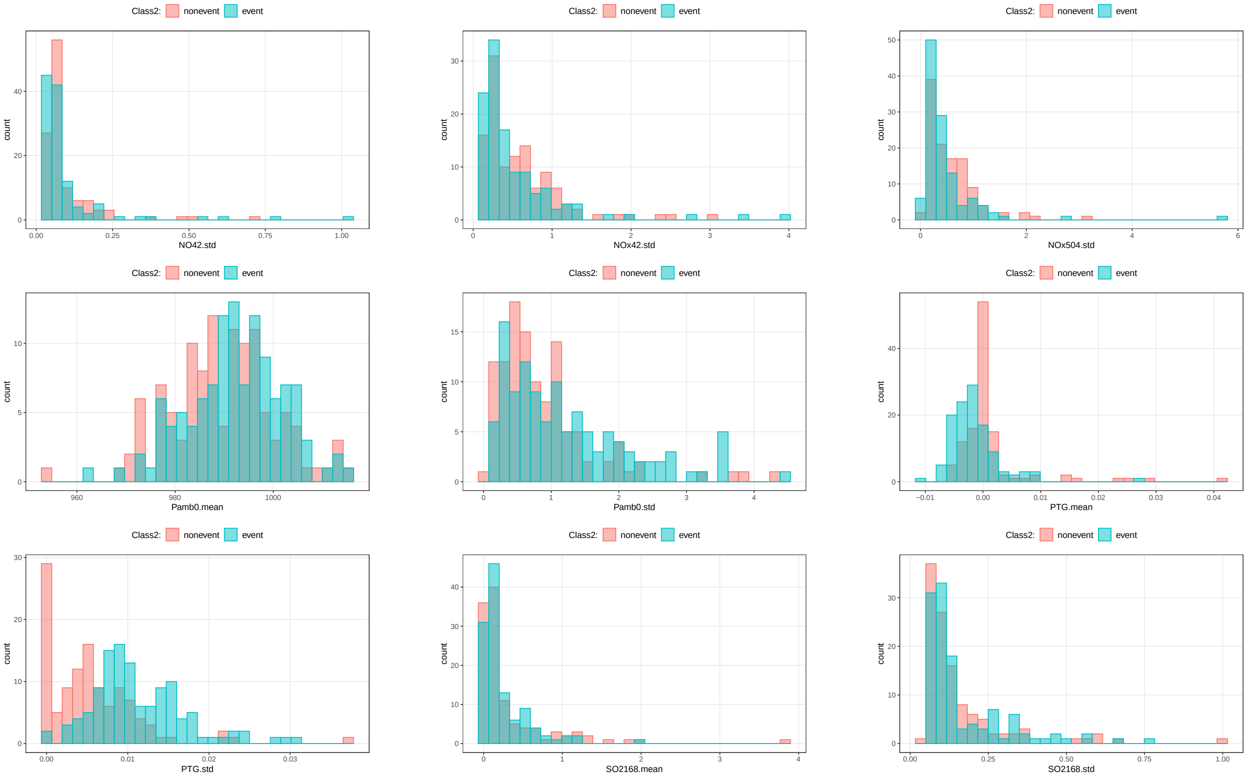
accuracy	perplexity
0.531	2.874

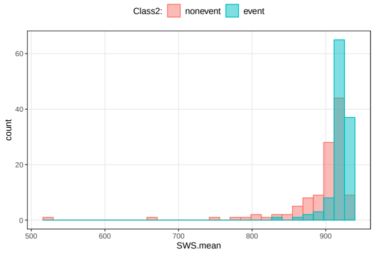
Compared to the performance in validation and cross-validation approaches, the performance has reduced significantly. The performance on testing data indicates that the method has no real prediction power. With this performance we should check the data process and feature selections again and try some other methods, too. For example, support vector machine with radial kernel (gamma = 0.5, cost = 0.1) gives promising performance measures on testing data:

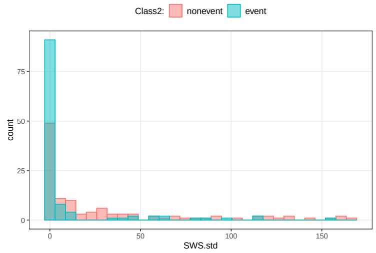
accuracy	perplexity
0.824	1.887

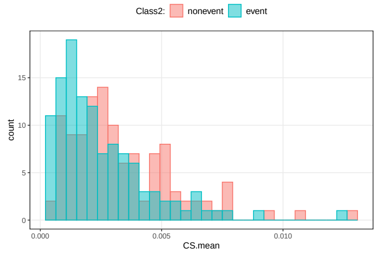
Annex: Histograms of variables

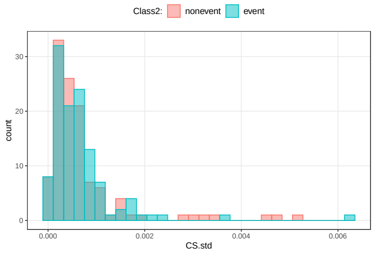
After removing variables with high correlation to other variables 26 variables are left in the data. The histograms give an overview of the values of the variables:

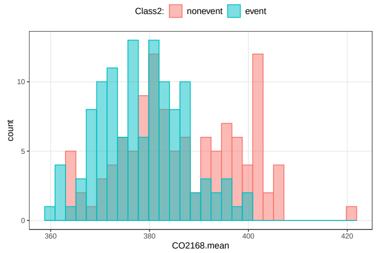


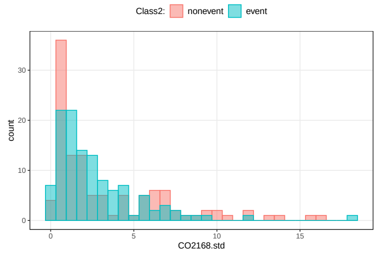


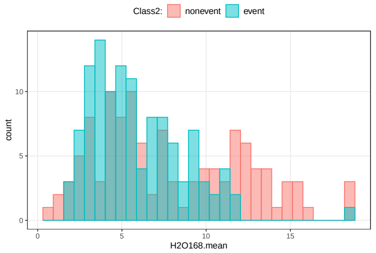


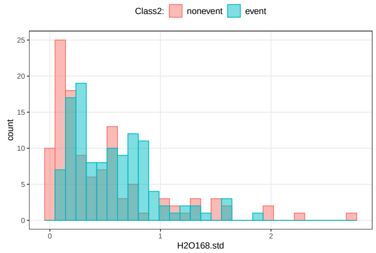


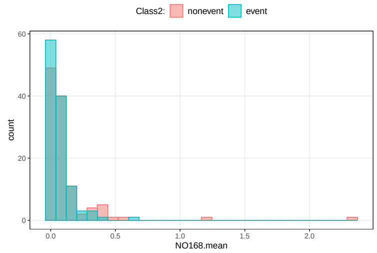


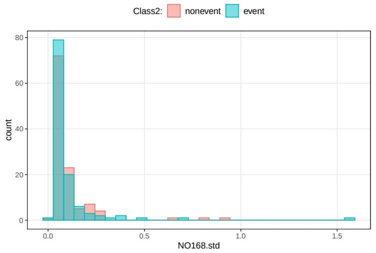


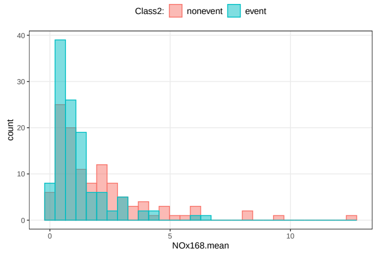


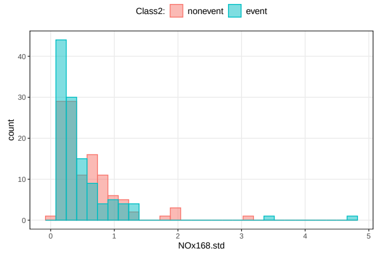


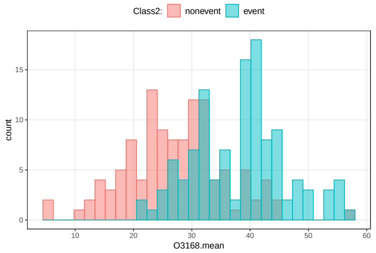


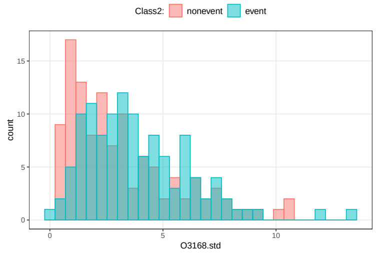


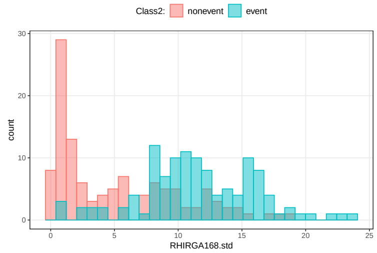


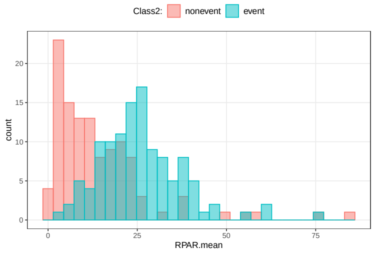


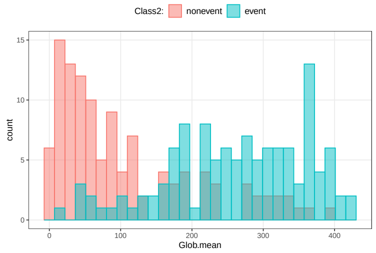












Grading section

Grade for the deliverables: 4

We have showed a good understanding of the topics. We have proceeded in soundly manner from data exploration to data cleaning, data analysis and investigated a variety of different methods on the data. We have used validation and cross-validation approaches to get a good, overall understanding of the performance of the methods and generally accepted criterion to select the classifier. However, there seems to be some shortcomings or missed insights since our binary classifier didn't succeed in the challenge. The results from the challenge and some video pitches gave us ideas to improve our work but unfortunately it was too late to do amendments. Our report shows a good quality: it's readable and written in coherent manner. The processes and selection done are clearly argued so that the reader can critically evaluate them.

Grade for the group: 5

We had a great group! Discussions were analytical, and we helped each other to learn the topics and do the exercises as well as the project itself. Everyone had a freedom to give an input suitable for his/her own situation. We did the project in time and with good quality. We all brought input from our own experience which completed each other. Thus, all were also able to do the parts of the project corresponding to own strengths. We all took responsibility of the project and wanted to help each other. We had fun with each other!