Identification of fire periods from air quality monitoring network measurements ML4Science project with LAPI lab

J. Biefer, A. Korukova, A. Mellinger
Department of Computer Science, EPF Lausanne, Switzerland

***Abstract*—Detection of smoke is a sophisticated problem which requires the high-costly equipment measuring the chemical composition of the samples. The goal of the project is to build ML models predicting the chemical composition from an alternative method, namely Fourier Transform infrared spectroscopy (FTIR). We tried linear machine learning methods to predict if a burning event occured. Feature selection has been performed using PCA and another method has been tested as well. Both regressors and classifiers were evaluated as well as a multi-layer perceptron. The best performance was found using logistic regression.**

I. Introduction

In order to keep a certain quality of life (or to improve it), the air is monitored for detecting peaks of pollution or changes in the composition of pollutants. For this purpose, many stations, collaborating together, regularly collect air samples and perform analysis to be able to determine the cause of the pollution. Amongst these analysis methods, one can mention smoke detection using satellite pictures or air analysis using expensive however very precise machines.

The LAPI lab is working on another technique using Fourier Transform infrared spectroscopy (FTIR) to reduce costs while still having satisfactory results. Measurements are made and the result is a range of values for a set of wavelengths (wavelength absorbance). Some computations using these values could lead to various conclusions and allow to detect fire periods.
This project aims at predicting, from this wavelength absorbance, if a fire is amongst the causes of pollution. In order to do that, we tried to predict the presence of smoke from our samples.

II. Data set

We have been provided with two types of spectra: the raw one and the baseline corrected, in which the contribution of light scattering from the filter membrane and absorption from Teflon filter C-F peaks is subtracted. These files contain the measured absorbance per wave number (11138 wavenumbers in total, ranging from around 400 to 4000 cm^{-1}) for each measure, and constitute our samples.

The values/labels we aim at predicting are given in the Ion Chromatography (IC) files. The prediction on the presence of smoke in the area is based on the following ratio: density of Levoglucosan over density of Water-Soluble Organic Carbon (WSOC). If this quantity exceeds 0.01, the sample is labelled 'yes': smoke is considered to be present. If it is below 0.005, no smoke is considered to be present. Otherwise, the sample is labelled as 'maybe'.
IC files, among other information, contain the levels of Levoglucosan, WSOC, their ratio and prediction.

To build the data files used for training and testing of our models, spectra files were combined with IC files. The resulting data sets contain 307 rows (spectra, samples) from which we finally obtain 304 (by removing some 'nan' values) and 11138 columns (wave numbers, features). The labels are not equally distributed in the data: around 65% are labelled as 'nan' (no smoke), 25% are labelled 'yes' and 10% are labelled as 'maybe'.

Note: other data sets were also provided, but weren't used.

III. Models and Methods

Packages

We used *sklearn* for pre-processing and machine learning algorithms (and related), *matplotlib* and *Seaborn* for visualization.

Predicted quantities

This problem is essentially a classification one. An important question is the status to give to the samples labelled as 'maybe'. These labels correspond to samples that could potentially go both ways. First, we have decided to consider all of the 3 categories. But then, is incorrectly predicting a 'maybe' label really an error? This prompted us to switch to a binary problem. The labels are based on the comparison of a physical quantity against two thresholds (one to label 'maybe', one slightly higher to label 'yes'). Thus, we can decide on the threshold for binary classification. For example, we could use an average of two initial thresholds. In most cases, we also figured it made sense to directly predict the Levoglucosan/WSOC ratio (continuous quantity), since training on labels could lead to loss of information. Predicted labels would then be retrieved by comparing to the threshold.

Used data set

We used both raw and baseline-corrected spectra to gain insight from the comparison of the results. We evaluated the models on standardized or non-standardized data since the results can differ depending on the context.

Measuring accuracy

The most simple way to evaluate the performance of the models is computing the ratio of correctly labelled samples - the "accuracy". But since there are much more 'nan' labels than 'yes' ones, a classifier predicting systematically 'nan' performs well using this criterion. To avoid this problem, we built confusion matrices and considered F1-scores for binary classification.
We consistently used cross-validation to overcome the small size of the data set: on some small test samples with almost only 'no smoke' labels, one might get artificially good results.

Linear models

It is a usual practice in chemometrics[1] to stick to linear models for their simplicity of use and because they usually perform well enough for the needs of the field. Also, the small number of available samples encouraged us to use rather simple models to avoid over-fitting.
Therefore as a first step, we tried a few basic linear models in order to get a sense of which one performed best. We mostly used ridge regression, ElasticNet and logistic regression as a direct classifier.

Note that the ratios we are trying to predict seem to come from an exponential distribution, with their logarithm following a uniform distribution (see figure 1 below). Therefore, it makes more sense to train linear models on the logarithm of this ratio.

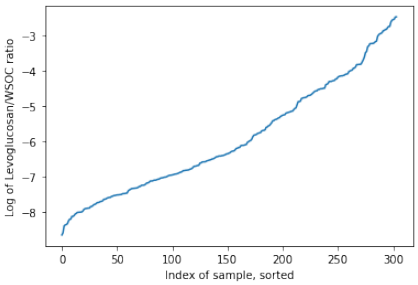

Figure 1. Log of sorted raw data

Considering the fact that we have about 35 times more features than samples, linear systems are poorly conditioned and have too many possible solutions. Moreover, an algorithm such as gradient descent in logistic regression takes a lot of time to converge for a high number of features. To solve these problems, we selected a small subset of 'good' features. To do that, we based ourselves on ElasticNet regression coefficients using the standardized data set. For the right parameter choice, ElasticNet indeed tends to decorrelate features and to only select a few of them.
To diminish the number of features, we also used Principle Components Analysis (PCA). It also provided us with an estimate of how many decorrelated features actually described the data.

Neural networks

For the sake of completeness and in front of the results we had with linear models, we used supervised learning to see if we could reach a better accuracy. We chose a Multi-Layer Perceptron Regressor with Limited BFGS as optimizer (popular for datasets with few samples) and ReLU as activation function to predict the value of the aforementioned ratio. The data provided to the Neural network is pre-processed and re-dimentionned using PCA. The model was tested with one and two hidden layers for various number of nodes per layer.

IV. Results

Unless stated otherwise, all results in this section are obtained using 4-fold cross-validation.

Linear models: 3-categories classification

Because of the non-linearity of the predicted quantity, training on this quantity itself gave an artificially large number of 'maybe's and bad results. We therefore replaced ratios by their logarithm. With these new values, the models predict about as many maybe's as expected, but they now consistently predict them badly. Below is the table representing accuracy for 3 different models and the confusion matrix generated with ridge regression (figure 2).

	Ridge	ElasticNet	Log. Reg.
accuracy	0.743	0.724	0.795

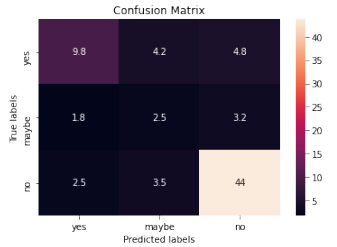

Figure 2. Confusion matrix for the ternary problem using Ridge, averaged overs 4-fold cross-validation. Maybes are badly predicted.

[1]"Chemometrics is the science of extracting information from chemical systems by data-driven means.", Wikipedia, Dec. 5^{th} 2020.

Linear models: 2-categories classification

The following table presents accuracies and F1-scores obtained for different models in 4 cases: using raw or baseline-corrected (b.-c.), standardized (std) or non-standardized data. We used a threshold of 0.0075 to label the samples in two classes. Results are slightly worse for a threshold set to 0.005 or 0.01.

	b.-c.	b.-c. std	raw	raw std
Linear reg.	0.766 0.538	0.776 0.543	0.779 0.550	0.750 0.488
Ridge reg.	0.776 0.382	0.832 0.539	0.789 0.417	0.806 0.483
ElasticNet	0.713 0.060	0.799 0.439	0.740 0.232	0.769 0.376
Log. reg.	0.799 0.444	0.908 0.629	0.789 0.445	0.852 0.577
SVM	0.779 0.400	0.769 0.410	0.750 0.304	0.756 0.305

Standardization usually yields better scores. So does the baseline-corrected data. Only linear regression seems to be an exception to that rule.
Overall, logistic regression gives the best results: below is a confusion matrix of the results it gives. Ridge regression also provides an interesting performance. It has the advantage of using and predicting continuous data, and it runs faster.

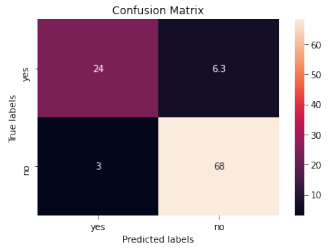

Figure 3. Confusion matrix for the binary problem using LogReg, averaged overs 3-fold cross-validation.

Feature selection

Feature selection based on ElasticNet coefficients does not improve scores. However, the scores do not decrease much when replacing the features by the 30 best ones selected on the basis of ElasticNet coefficients. In terms of accuracy, we lose 0.02 for ridge regression, 0.04 for logistic regression. In terms of F1-score, we lose 0.03 for both. Nevertheless, their use makes logistic regression converge way faster.

PCA provides more interesting results. The following table represents how much of the total variance is contained in PCA's first n components.

nb. of components	1	5	10	30
baseline-c.	0.49	0.85	0.95	0.997
raw	0.63	0.97	0.997	0.99996

From the table IV above, we note that baseline-corrected data contains more information than the raw one, in the sense that it requires more principal components to include the same quantity of information. We lean on this result to give a preference to baseline-corrected spectra. Their richness compared to the raw ones is illustrated in appendix with the comparison of both scatters of the data for k = 2 principal components.
With 30 components, logistic regression gives the same results as with the whole data set. Ridge regression loses 0.02 accuracy, and 0.05 F1.

Neural network

We iterated over the first 100 nodes for the first and single hidden layer to find the combination that maximizes the accuracy. The results highly depend on the initialization of the weight and bias (random_state value on *sklearn*) and can change up to 58% from one run to another. By averaging over multiple different initialization values, the neural network only reaches 60.5% accuracy at its maximum with 60 nodes on a hidden layer. The mean accuracy is at 40.3% and in the worst case the model goes down to 25.7% accuracy (see Jupyter Notebook with full results).
We also tried to get results and to find the maximum accuracy with two hidden layers but it was only worse.

V. Discussion

Regression vs. classification

Detecting smoke cannot be reduced to a discrete nor continuous problem only. The decision is made on a continuous measurement so one could try to predict the ratio with regression and label it according to the thresholds. But the decision is a discrete problem that could be solved with (ternary or binary) classification. This removes the need for threshold selection (even if used to label the training set).

At first sight, it may seem surprising to get a better performance with logistic regression than with the models trained to predict the real values. One could argue that information is lost when labels are used instead of the values from which they are derived. This is true only to some extent. For example, take two samples with Levoglucosan/WSOC ratios equal to 10^{-7} and 10^{-4}. The models predicting continuous values will view these ratios as quite different, whereas the binary classifiers will see them as similar, since both yield the 'nan' (no smoke) labels. In this particular case, making no distinction between these values might be closer to reality. In other cases however, the distinction brought by regression is useful - spectra labelled 'maybe' for example.

It should be noted that ridge regression still gives pretty good results. ElasticNet, from which it is a particular case,

performs worse because it is very hard to optimize with so many features.

Ternary vs. binary classification

In the log scale that we used, labels values range from -9 to -2.5. The threshold for a 'yes' is only $\log(2) \approx 0.7$ above the threshold for a 'maybe'. Unless we have a very precise estimator, predictions for 'maybe' are doomed to being bad. This along with the fact that we do not care so much about knowing whether there 'maybe' was smoke, is why the binary classification makes more sense.

Putting the threshold for the 'smoke impacted' label halfway between the thresholds used for 'maybe' and for 'yes' works best, for we want to make the amount of falsely labelled samples given to the model as small as possible.

One should keep in mind that using binary labelling implies to make a choice for the samples named 'maybe'. This choice might be wrong in some cases: that way, labels can not always be taken as an absolute reference, which limits the accuracy we can obtain.

Performance of feature selection

Reducing the number of features is mostly important to make optimization algorithms run faster. We were surprised that it didn't also improve the performance. A 304×11138 linear problem such as ridge regression has too many solutions, and even when working with standardized data, it seems likely to generate overfitting. We took a close look at the *sklearn* methods we used and could not find any built-in feature selection methods.

The pertinence of using ElasticNet regression coefficients can be evaluated by confronting its coefficients for predicting Levoglucosan and WSOC with the plot of absorbance for a clearly smoke impacted sample, for example sample 118 of our data set: measures made at station TALL1 on 2008/04/03. The two graphs below show interesting correlations. For example, Regions around 800-900 cm^{-1} are those used by ElasticNet to predict levoglucosan, and also yield high absorbance for this sample.

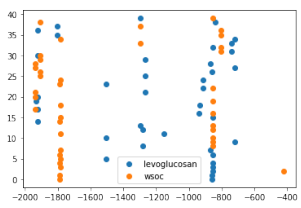

Figure 4. Wavenumbers with highest weight to predict Levoglucosan and WSOC

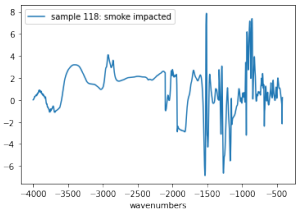

Figure 5. Spectrum of sample 118

Neural Network

The prediction of a neural network is not great, as expected (and warned by the lab mentors). After trying to feed to model the whole dataset (without feature selection), we noticed the need to reduce the dimensionality since the neural network required a lot of iterations to converge to poor results. Selecting features using PCA resulted in faster convergence and slightly better results but still not sufficient nor better than the linear models we encountered earlier. The considerable difference in the predictions of the network is certainly due to the very small training set at its disposal. Increasing the number of hidden layer (to two) does not improve the prediction. Indeed the core issue stay the same.

Other attempts

Instead of predicting the log of the ratio Levoglucosan/WSOC, we also tried to predict each of these two independently. The results were slightly worse than when directly predicting the ratio, probably because errors for each component sum up.

VI. Summary

In this project, we tried to build the best classifier for declaring a sample smoke impacted. We used a wide range of different models on different data sets to find out that logistic regression performed best on a binary classifier with a carefully chosen threshold. Using the 30 principal components of the normalized baseline-corrected spectra, we found an accuracy of about 90% with F1-score 0.62 and small running time of the algorithm.
Overall, it seems that linear models best suit the problem.

VII. Appendix

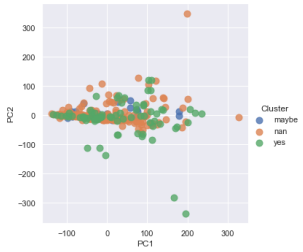

Figure 6. 2 principal components: raw spectra

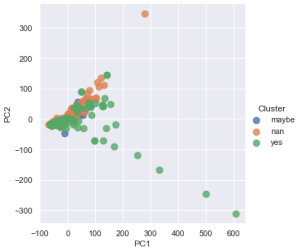

Figure 7. 2 principal components: baseline-corrected spectra. Some smoke impacted samples are highly visible

Acknowledgements

The authors thank M. Satoshi Takama and M. Amir Yazdani of the LAPI lab at EPFL for their explanation and guidance through this project.

References

The EPFL Machine Learning class 2020 from Martin Jaggi and Nicolas Flammarion.