Plume Labs data challenge : predict pollution in an Asian megacity

Wiem Gharbi and Hugo Vallet

Abstract— **This report describes our participation and the solution we proposed for the "Plume Labs data challenge". It was part of the "Advanced machine learning" course of the Ecole Polytechnique led by Mr E. Le Pennec and Mrs F. D'Alche-Buc.**

I. INTRODUCTION

Plume Labs is a french start-up aiming at providing people insights about the pollution in the atmosphere. In particular, they use pollutants measures gathered from international meteorological agencies to compute a score, the plume labs "air quality index" in different cities in the world. This score takes into account multiple pollutants and toxic gases such as fine particles, ozone or nitrogen dioxide.

II. TASK OVERWIEW

Presently, the start-up is able to gather large amounts of data and process them in order to compute the quality index in real time. They now want to use this expertise to make predictions of the amount of pollutants in the future. More precisely, they want us to predict the concentrations in the atmosphere of 4 different pollutants during the next 24 hours.

In order to achieve that, they gave us measurements from different meteorological stations in an Asian megacity. The set of measurements aggregates the measures of 18 stations. There are 2 categories of stations :

- The first category is composed of 17 stations having measured pollutants levels during the previous 24 hours.
- The second category is composed of the station on which we want to make the prediction. For this station we have no pollutant data but we are given the meteorological features (humidity, dew point, cloud cover, temperature, etc.) of the previous 24 hours as well as weather forecasts for the next 24h.

Thus, we have to implement the classical machine learning pipeline, summarized by the following graph :

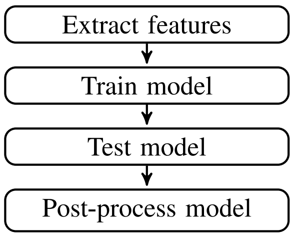


Fig. 1: The process to be implemented

A. Extract features

The data we are given is already "clean", there are no missing values or strange values. Moreover, the number of features provided is large: we have 3728 different features in the data set. Thus, we focused this part of the process on understanding the features and also on rearranging the features to train different models.

B. Train model

Here is the most tricky part of the challenge. We are facing a multi-regression problem : we want to predict the levels of 4 different pollutants for the next 24 hours. In other words, we have to be able to perform, for each example in the train or in the test sets, 96 regressions. Moreover, since this is a time-related prediction problem, we have to be able to include in our models the time information.

C. Test model

To test our models we used classical cross-validation on sub-samples of the initial data. The quality of the prediction was assessed using the Mean Squared Error (MSE).

D. Post-process model

Keeping in mind the "shape" and characteristics of the data on which we want to predict was fundamental in this challenge. After predicting, we used shape considerations to post-process the predictions in order to increase our accuracy.

Finally, after finding a good predictive model on our local set, we made submissions on the Ecole Normale Superieure data challenge platform (https://challengedata.ens.fr) on which the challenge was hosted.

III. ANALYSIS OF THE DATASET

At the beginning of the project, we tried to understand better the given dataset performing different analysis on the data.

A. Description of the data

The data is organized in a very specific manner. When we import the dataset we get the following matrix :

$$\begin{array}{cc} & \begin{array}{ccc} hour_{-24} & \dots & hour_0 \end{array} \\ \begin{array}{c} day_1 - hour_1 \\ \vdots \\ day_N - hour_{24} \end{array} & \begin{pmatrix} mes_{1,1} & \dots & mes_{1,P} \\ \vdots & & \vdots \\ mes_{N,1} & \dots & mes_{N,P} \end{pmatrix} \end{array}$$

Fig. 2: This is the representation of a feature-block containing the measures. Each measure is made on 25h, thus for each pollutants and for each stations we have blocks composed of 25 columns : one column per hour. The entire dataset is just a vertical concatenation of that kind of blocks.

We also noticed that, and this is really important, from a line to the following, the measures are made with a 1-hour time interval. Because of that, we observed ”propagating” values in the data set, and we took advantage of this information latter in the post-processing.

B. Features available

The given data is heterogeneous : it’s clear that we have two categories of stations (described before) but even among the pollutants-describing stations, there are differences. Some of them are measuring just one or two types of pollutants while some others measure all pollutants... For that reason, we do not have the same quantity of information for all the pollutants.

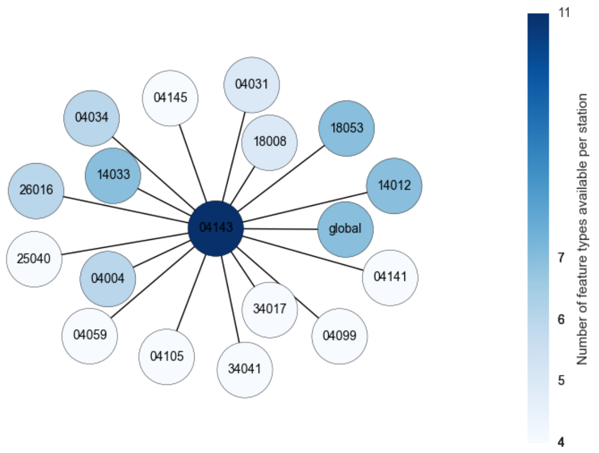


Fig. 3: Here are listed the stations (identified by their id numbers) contributing to the final prediction we are going to make on a station named ”04143”. 04143 is the station with the highest number of features because it contains all the meteorological data and forecasts. ”Global” is not a physical station, it contains some additional non-meteorological data. The other measure the pollutants levels and wind. From that graph, it appears clearly that the richness of information from a station to another is not the same.

An other way to understand this is to look at the total number of features per station or the total number of features per type of features :

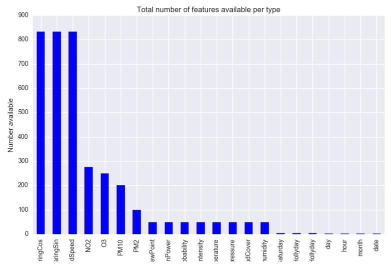


Fig. 4: Every station except ”global” have wind measures but only few stations have PM2 measures,for instance.

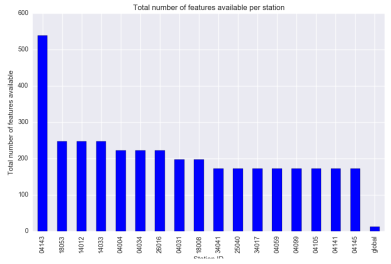


Fig. 5: One station has a lot more information than the others : it’s the station on which we are asked to make the prediction. This is quite understandable since this specific station measures a lot of meteorological features, in general on 24h or 48h time windows

C. Correlation between features

Because of the specific distribution of columns in the dataset, we wanted to check the correlation between the input features (columns of the training dataset) and the correlation between the features we want to predict (observed values used for training).

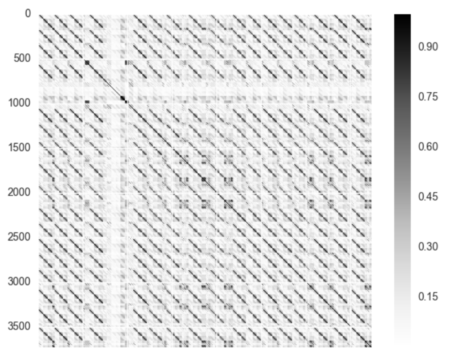


Fig. 6: Here, each pixel value, p_{ij} is the correlation between the column i and the column j. The main phenomenon observable here is the presence of multiple small diagonal lines of high-correlation. They are due to the wind measures. The wind measure at an hour h in a station s_x is always highly correlated to the wind measured at the same hour in a station s_y and this is true $\forall x, y$.

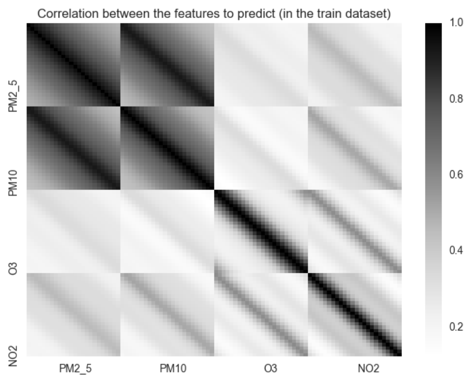


Fig. 7: Here is the same matrix than before but, here, calculated on the 96 columns we want to predict in the training set. First, we can observe that we have 16 squared sub-blocks : they correspond to the columns of different pollutants. In each sub-block, the values observed measure the correlation of a pollutant at hour h_x with itself at hour h_{x+t} : all pollutants are highly correlated to themselves on close hours (black diagonals) but the correlation diminishes when time increases. This is quite understandable. Secondly, the fine particles (PM_{10},PM_2) are less volatile than gases (NO_2, O_3) : their levels of self-correlation are higher. Finally, PM_2 and PM_{10} are obviously highly correlated but this is less true for gases.

From this information, we adopted the following approaches:

- We have to test multi-models, that is to say models based on the observations of just one pollutant predicting the same pollutant or models trained on all observations but predicting, again, on pollutant only.
- We have to feature engineer the wind features. They are highly correlated and this characteristic can perturb our models (and increases the training time for no additional value).

D. Behavior of the pollutants concentration

Finally, we also checked the pollutants concentrations' behavior. From the train set it is easy to track the pollutants in each station hour by hour : in the dataset each column correspond to an hour of measure. Thus, we arbitrarily took the column corresponding to the "hour 0" of each station for each pollutants and plotted the evolution.

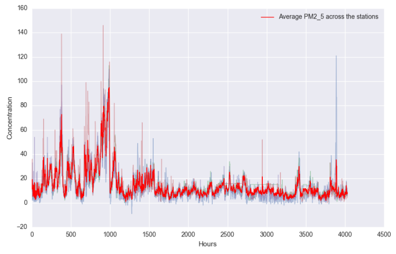


Fig. 8: In the PM2 levels we can see a relative stability from an hour to another. The deviation in the measures between the stations is low.

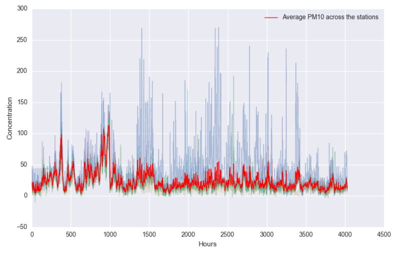


Fig. 9: As you can see, the levels observed for PM10 are close to PM2's ones. We also notice the presence of an outlier station (blue curve) for which the measured values are often very far from the mean value.

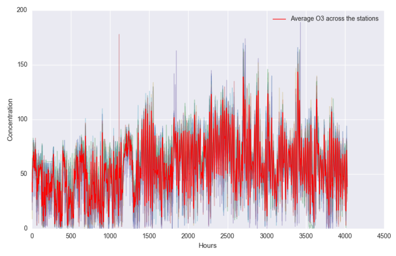


Fig. 10: Clearly, we can verify the volatility of O_3 that we intuited from the second correlation matrix. From an hour to another we see huge variations. Nevertheless, the measured values, for each stations are close to the "mean" value.

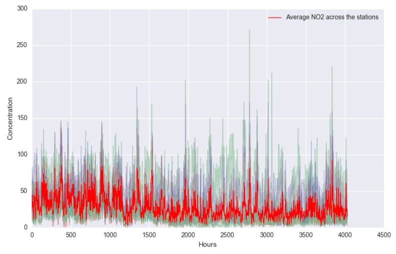


Fig. 11: Finally, here are the observed values for NO_2. We find again a high volatility and also an important deviation between stations.

IV. First Approach

A. Ridge and Lasso

After analyzing the dataset and having a better understanding of the task at hand, we started experimenting with the regression task. Our first approach was to use a linear model, more specifically a **ridge regression** which consists in minimizing linear least squares function with an L_2 penalty. The L_2 penalty controls the complexity of the model through regularizing the weights to keep them small, thus controlling over-fitting.

The intuition behind using the ridge regression was mainly driven by the fact that our problem is quite high dimensional. In fact, we have 3729 features in our data-set. We chose Ridge over Lasso for a very simple reason: Some of our features are highly correlated (such as hourly measurements of PM_2 and PM_{10}). If we use L_1 penalty, if two features are correlated, in order to ensure sparsity of the solution, only one of the two features will be picked. In contrast, if we use L_2 penalty, the two correlated features will be both included in the model and their corresponding coefficients will be shrinked. In the case of both regressors, the penalization hyper-parameter C was selected using cross-validation. The obtained hyper-parameter was C=0.1 in both cases.

	MSE
Ridge Regression	85.12
Lasso Regression	130.46

TABLE I: Mean cross validation score using MSE on the training dataset

B. Random Forests, Gradient Boosting and Feature Importance

Our second approach to the model was to use ensemble methods. Since we have some noisy data in our training set, we came to the conclusion that we need a more robust model. We therefore opted for ensemble methods.

Using random forest and gradient boosting regressors we build the following model: For each hourly target value of a specific pollutant, we build an ensemble regressor which takes as an input the entire train dataset. We hence obtain 24 models for each pollutant amounting to 96 models in total.

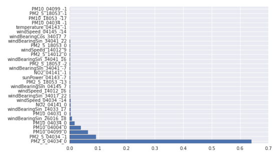
Using this approach, we manage to improve the performance of our model. To better evaluate this performance, we use on top of the mean squared error the R^2 or "R-squared" score which stands for coefficient of determination. R^2 measures how well are the target data replicated by the model. The R^2 score is computed as follows:

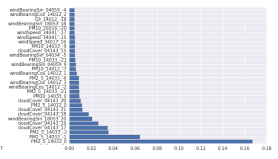
$$R^2 = 1 - SS_{res}/SS_{tot}$$

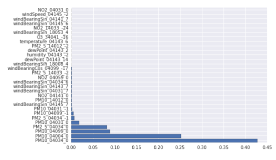
$$SS_{tot} = \sum (y_i - \overline{y})$$

$$SS_{res} = \sum (predy_i - y_i))$$

Where $\overline{y}$ is the mean of the target values and $predy_i$ are the values predicted.


(a) PM_2-04143-1


(b) PM_2-04143-24


(c) PM_{10}-04143-1

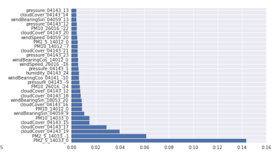
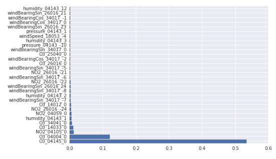
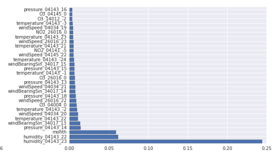
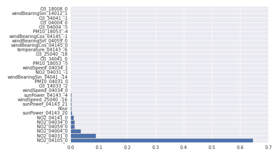

(d) PM_{10}-04143-24

Fig. 12: Feature Importance of the Random Forest Regressor model used to predict PM_2 and PM_{10} res. at hour 1 and hour 24.


(a) O_3-04143-1


(b) O_3-04143-24


(c) NO_2-04143-1

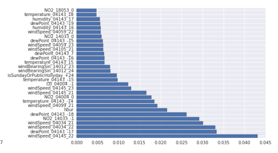

(d) NO_2-04143-1

Fig. 13: Feature Importance of the Random Forest Regressor model used to predict O_3 and NO_2 res. at hour 1 and hour 24.

	R2
Ridge Regression	0.72
Lasso Regression	0.57

TABLE II: R2 score using a train-test split on the train dataset

In Fig-12 and Fig-13, we present some of our results regarding the feature importance that we have obtained with the Random Forest Regressor Model: As expected, features of PM_{10} have an impact on the prediction for PM_2 at the hour 1 and, conversely, features of PM_2 have an impact on the prediction of PM_{10}. We have also noticed that features of NO_2 have an impact on the prediction of O_3. The inverse effect is not as marked.

As demonstrated by the feature importance graphs in Fig12-b/ and d/ and Fig13-b/ and d/ the coefficients of features tend to decrease when we have to predict the levels of pollutants for the later portions of the day (hour 24, 23, 22...), which means that the features are less significant. This phenomenon is quite understandable as we are predicting the levels of pollutants of the hour 24 based on measurements of the previous day, which is a harder task than predicting the level of pollutants for the hour 1 based on measurements of the previous hours.

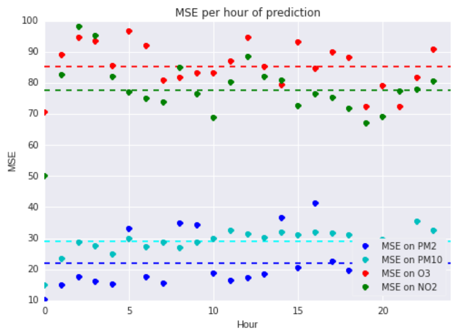


Fig. 14: Mean squared error on each of the 96 prediction columns. The model used is a Random Forest Regressor with 30 estimators and maxdepth of 6

In Fig-14, we notice that the mean squared error (mse) on PM_{10} and on PM_2 is much smaller than the mse on NO_2 and O_3, which validates our observations on the high volatility of the latter. We also notice, that the mse on the first column (prediction for hour 1, corresponding to the first dots of our graph) is significantly smaller than the mse on the other predicted values.

C. *Post-processing with predicted values propagation*

If you read carefully the section "Analysis of the dataset" you know that the given dataset is vertically **and** horizontally ordered by hours. Thus, we observe in the dataset "propagation" of constant values. In other words : the values on the inverted diagonals of the matrix of prediction are constant because they concern the same hours of prediction !

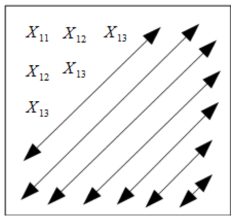


Fig. 15: For each pollutants measures, you observe this pattern of propagating values on inverted diagonals. This is caused by the order of the set of examples.

The data on which we want to predict is organized exactly in the same way. This is an important piece of information as we know, now, that the predicted matrix must show a propagating pattern. Now, if you combine this information with the fact that predicting the first hour of each pollutant levels (that is to say predicting the first column of each pollutant) has a smaller mse (ref. section 4/ B-Random forest and feature importance) than predicting the next hours then you obtain the following method :

- **Step 1**: Predict the first column of each pollutant as precisely as possible, for example using gradient boosted trees with a large number of estimators.
- **Step 2**: Predict the other columns with a constantly decreasing number of estimators to gain time. The quality of the prediction of the other columns is, then, a lot diminished, but we do not care because of step 3.
- **Step 3**: Post-processing of the prediction : propagate the values from the firsts columns in the way displayed on fig-15. Note : when we propagate we simply erase the predictions made before with newer values from the firsts columns.

Apart from improving our scores a lot, this post-processing allowed us to decrease the computational cost of the training and, thus, dedicate that extra time to train more complex and more precise models for predicting the first columns.

D. *Conclusions of the first approach*

- On this dataset, gradient boosting regression trees are the best learners we found (they are, in fact, slightly better than Random Forests).
- Training one model per column is clearly a better way to address the problem.
- The quality of our prediction depends of the pollutant and the predicted hour: while PM_2 and PM_{10} levels can be predicted with high confidence, volatile gases like NO_2 and O_3 are more difficult to fit.
- This 96 regression problem can be re-interpreted as a 4 regression problem: the order of the data set allows us to propagate predicted values in a way such that we no longer need to train one regressor per column.

V. IMPROVED MODELS

From the previous section we understand now that all the challenge resides in making the prediction of the first columns of each pollutants as precise as possible (and the propagate the values to recover the prediction matrix), especially NO_2 and O_3's ones. This section describes the last improvements we made on our pipeline.

A. *Hyper parameters optimization*

After concluding that gradient boosting regression was one of our best performing solutions, we wanted to optimize the hyper parameters of the model. As a first approach, we have used a Grid Search with Cross Validation to tune the parameters of our model. We train the regressor using the library XGBoost which gave, on this dataset, similar performances to sklearn's, for a quicker training time. Moreover, this library combined with the powerful hyper parameter optimization tool "hyperopt" gave us one of our top performing submission on the challenge site.

B. *Feature engineering*

As we saw on figure 6, in the training data, a lot of the features are highly correlated. This is particularly true for the features concerning wind measures across the stations. We tried the different feature engineering approaches :

- Aggregation of wind features : we replaced all the wind features (i.e. wind for each station) by their average value hour per hour. This allows us to remove around 2000 unuseful features.
- Average of the pollutants measures : we took the average of all the pollutants' measures, hour per hour, and added these new features to the dataset.
- Slop coefficient of the last hours : to add a simple feature coding the "volatility" of the evolution of the pollutants concentration, we computed, for each station and for each pollutants, the difference between their value at hour 0 and at hour -1. The coefficients obtained were, then, used as normal features during the model fitting.

While the 2 firsts approaches didn't allow us to increase our precision, the last one performed well on the gases. Here is a summary of the results we obtained :

	PM_2	PM_{10}	O_3	NO_2
No feature engineering	9.97	14.13	60.90	43.67
With slop coefficient	9.35	12.53	45.29	40.72

TABLE III: Cross validation score per pollutant using MSE on the training dataset. The fitted model was our "best model", i.e. a Gradient Boosting Regressor trained with XGBoost with the best set of hyperparameters.

C. *Bagging models*

Finally, we have used a method combining our best performing models, i.e. Random Forest Regressor and the Gradient Boosting Regressor. We have used an average of the predicted values of these models as a predictor. The aim behind this approach was to even out the individual weakness of each learner. Using this approach on our feature engineered training data set, we managed to increase the performance of our model both locally (using cross validation on the training dataset) and on the online platform.

VI. CONCLUSIONS

A good understanding of the task and data's characteristics was fundamental to perform well on the challenge. Using ensemble methods was also key, since the data lives in a high dimensional space with a lot of heterogeneity in the parameters measured. Despite our efforts, we didn't manage to decrease the error made on NO_2 and O_3 to the level we achieved for PM_2 and PM_{10}. Feature engineering, an issue we tackled late in the project, might be the key.