AML Lab Report

Challenge 1 - Weather prediction

Name of Student 1: Zachari Thiry
Name of Student 2: Matthieu Ramon
Name of Student 3: Benjamin Salon
Promo: 2023

AML : Advanced Machine Learning
(Spring, 2022)

EURECOM Graduate School and Research Center in Digital Science
Data Science Cursus

11th May 2022

Abstract

Machine learning exercise made during the Algorithmic Machine Learning course at EURECOM.

Based on a collection of different measurements from all over the globe we predicted the temperature at different locations as a regression problem. We first analysed the data to get the most exploitable features and transformed the data set with principal component analysis. Then we augmented the data set to take in account some useful computed features such as the hour of the day or the month. Finally, we selected and tuned different models such as a XGBoost regressor or a CatBoost model.

Contents

1 Data Analysis

1.1 Introduction to the data set

The given data set is the gathering of three forecasting services *namely gfs, cmc* and *wrf* alongside factual data such as "time", "location" ... The total number of available data points amounts to 1.9 millions.
The data set is unprocessed when received. As such, the types and the feature ranges are very heterogeneous and will need pre-processing.
The output we seek to predict is the "fact_temperature" value for each data point.

1.2 Analysis and first thoughts

We start by looking at the features thanks to the kaggle feature statistics viewer.

- **NaN and outliers:** We can see that we have 111 features from three different forecasting services. For each service we have a boolean feature which informs on whether the data from the corresponding service is available or not for this data point. If the data is unavailable for this data point the rest of the features for this service is NaN. We also have a feature gfs_soil_temperature_available which inform inside the gfs service if the soil temperature value is available. If not, the feature gfs_soil_temperature feature will equal an outlier value of -10000.
- **Feature exploitability and correlations:** By looking at the plots "fact_temperatures" in function of the different features we can select some features for their direct relationship with the output we want to predict. We selected 16 features that seemed directly correlated to the temperature. Here are some examples:

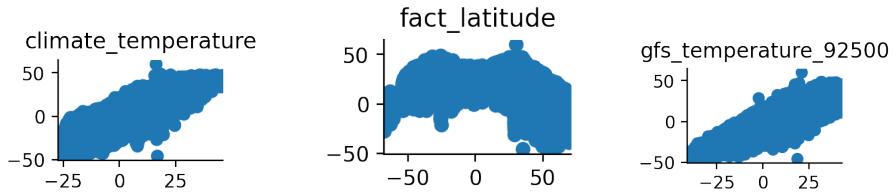

Figure 1: Temperature in function of climate_temperature, fact_latitude and gfs_temperature_92500

2 Data Pre-Processing

2.1 Augmentation of the data set with the sun hour and the month

Following the analysis of the data set, and considering that the target prediction were temperatures, we were startled not to see either the seasons or

days of the month passed as inputs. As such, we started by augmenting the data set with two columns extracted from the timestamp : hours and months. Review of the dependence between hours, months and factual temperature indicated good correlation with months, and bad correlation with hours. This is not a surprise. The computed hours of the day are GMT time and what we need is the actual time of the day for a given location. For this reason, we introduced *physical hour* which is computed from GMT time and a linear translation relative to its longitude : eg. an longitude of +90 induces +6 hours on the clock.

Immediate effects on a basic linear regression model was a 10% decrease in the RMSE going down from 2.5 to 2.2 . PCA analysis showed that for the ten first components' variance kept was increased by 9% once these two features were added.

2.2 PCA analysis as a transformation of the data set

In order to apply a feature reduction, we sought the use of the PCA algorithm. First however, the data should be **normalized** to make the PCA more effective *(see standardize_and_prepare() method)*. Let's note as well that some of the algorithms we tested later on were more known to be more efficient on standardized data sets. Normalizing it was thus a double necessity.

PCA analysis revealed that most of the data variance could be explained in just a small subset of the features. This fraction could be increased by about 9% if both hours and months were added to the data set. From 80 features kept and more, almost 100% of the variance was kept which can be seen in the graph below where basic algorithms don't perform any better given more than 80 of the most important features.

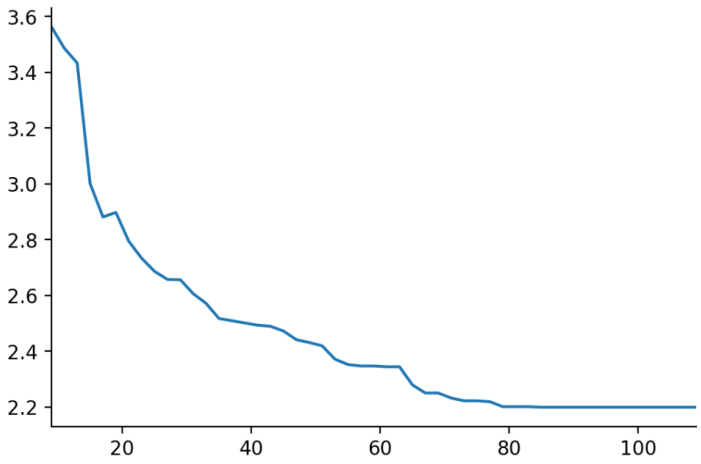

Figure 2: RMSE score of Lasso Regression in function of the numbers of components kept by the PCA (over the 111 components)

3 Model Selection and results

3.1 Choice of a boosting method

After a few different machine learning model tested (Lasso linear regression, Random forest regressor, Neural Network, Deep neural network) we oriented our research onto a boosting algorithm. The boosting and bagging method of the sklearn library such as the RandomForestRegressor were too long to fit the big dataset even with high factors of size reduction. After some research we chose to use the XGBoost regressor model and the CatBoost model because of their efficiency.

3.2 XGBoost

The XGBoost Regressor model is a model from the XGBoost library written in C++ based on *gradient boosted trees*.

After tuning the hyperparameters of the model (see section Model tuning below) we found as best parameters: 'alpha': 0.001, 'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 7, 'objective': 'reg:squarederror'

After submission on the Kaggle competition we reached a test root mean square error of 1.93 which constitute our best submission.

3.3 CatBoost

After some researchs we discovered another gradient boosting tree model, the *CatBoost regressor*.

We obtained a submission of 1.94 on Kaggle with this model but we lacked time to tune this model better. This model seems even more efficient that the XGBoost one and we could expect to have better results with a finer tuning of the model.

3.4 Model tuning

For each model we tried, we followed the grid search protocol. We first define the range of interesting parameters to tune and we then use the K-fold cross validation with all the different combinations to find the best combination.

4 Limits reached and self analysis

- **Model selection** We have found ourselves a bit in difficulty concerning the model selection process for this exercise. More precisely, we found some issues regarding the time of running which made the process of comparing model and tuning hyperparameters very tedious. We also encountered many environment issues with regular crashes of the kernel that can be explained by our lack of experience using the tools and lack of preparation.
- **Performance** For each model we tried, we couldn't go below the 1.93 - 2 of rmse score with many models converging at around 2.1 . We deduced that we may have lacked of a better feature analysis and data pre-processing. Even if data analysis and pre-processing represented a very important part

of our work, we expect to put even more emphasis on it in the coming challenges.

- **Organisation** This first AML project as a team work in data science was a real organisational challenge. We had to conciliate new tools with a new way of working as a group. We faced some difficulties on this side but we improved a lot during this first project and we expect to get better as we go on this part for the next two projects.