EURECOM

Sophia Antipolis

Algorithmic Machine Learning

Challenge 1 - Technical Report

Group 20

Professor: Pietro Michiardi

Authors: Daniele Falcetta
Simone Papicchio
Massimiliano Pronesti
Federico Tiblias

May 12, 2022

1 Introduction

The Weather Forecast problem consists in using pairs of meteorological features and target values at a particular latitude/longitude and time to predict the air temperature measurements at 2 meters above the ground. The dataset contains 112 features which are either direct measurements (such as sun elevation at the current location, humidity, temperature, pressure and topography, and other meteorological parameters) or weather predictions provided by climatic models (Global Forecast System, Global Deterministic Forecast System from the Canadian Meteorological Center, Weather Research and Forecasting). Each model returns the following predicted values: wind, humidity, pressure, clouds, precipitation, dew point, snow depth, air and soil temperature characteristics. Where applicable, the predictions are given at different isobaric levels from 50 hPa (20 km above ground) to the ground level. The features are highly heterogeneous, i.e. they are of different types and scales and are highly correlated between each other. Most importantly, this dataset is afflicted by a serious shift between train and test distributions.

Dataset shifting is a phenomenon that occurs when the distribution of the train and test data differs. This may cause several problems because the model is trained on one distribution but is used to make predictions on a different one, resulting in poor results. There are different types of data shifting such as Covariance Shift[1], Probability Shift[2] and Concept Shift.[3] In this work we analyze the presence of the Spatial Covariance Shift and try to address its main challenges. A visualization of the difference in distribution for the datapoints can be seen in Fig. 1.

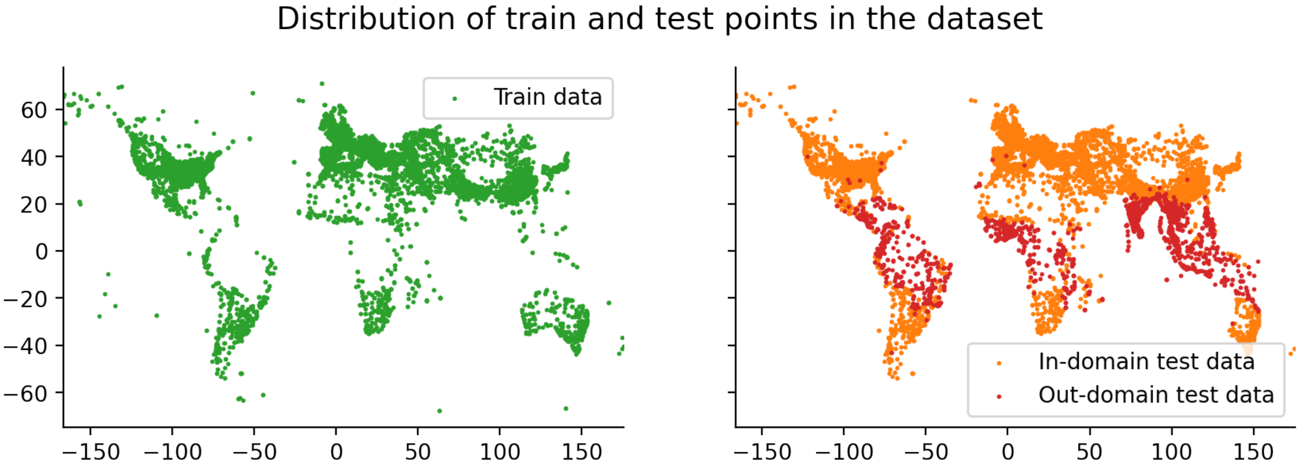

Figure 1: The figure clearly shows that the regions in the range of latitude from -18 to 10 are not present in the train dataset. This phenomenon is called *Spatial Covariance Shift*.

2 Data Analysis

2.1 Outliers

The first step of our dataset analysis focuses on outliers detection. Our approach, based on the Interquartile method (IQR), consists in plotting a boxplot for each feature. To obtain better and more organized visualizations, we plot four different graphs, one for each weather forecast model and, given the high heterogeneity of the features, we scale our data between 0 and 1 according to a min-max normalization. This analysis shows that almost all the features contain some outliers. In Section 3 we show some experiments attempting to remove part or all of them. A particular mention has to be done for the feature *gfs_soil_temperature*, which contains some entries equal to -9999 °C, i.e. a missing measurement, as reported in the associated feature *gfs_soil_temperature_available*.

[1] Changes in the independent variables or features of the dataset

[2] Changes in the target variable or the dependent variable in the dataset

[3] Change in the connection between the independent and the target variable across datasets

2.2 Correlations

Understanding the correlations among the features in the dataset is another crucial point since it can affect the results of some models. We start analyzing the correlation between the features and the target. Table 1 shows the ten features that correlate most with the target. An approximate analysis would lead to the removal of all the other features. However, this approach might be inaccurate: the Pearson coefficient only analyzes linear correlations, without taking into account non linear ones. In addition, we are not bearing into consideration that features might correlate between each other. If this is the case, two highly correlated features carry the very same information and one of them can be dropped. Conducting this analysis, we figured that the *gfs_temperature_xxxxx* features are in fact correlated when they are in similar conditions[4]. The same kind of correlation can be detected among other groups of features.

2.3 Feature Shift

Train and test datasets show a noticeable shift in the distribution of samples: test datapoints are distributed homogeneously around the globe, while the train dataset lacks samples in the equatorial region. To gain a deeper understanding of how this phenomenon affects our prediction, we compute a metric reminiscent of the Wasserstein distance. For each feature we divide the dataset into 100 bins of equal size and normalize them in order to obtain histograms. Then, we compute the area difference between the cumulative sum of these histograms for train and test datasets. The value we obtain represents some notion of distance between the two distributions and can be used later on during feature selection. Table 2 reports the ten most shifted features according to this metric.

Feature	Pearson Correlation
wrf_t2_interpolated	**0.963**
wrf_t2_next	0.958
gfs_temperature_97500	0.874
gfs_temperature_95000	0.863
climate_temperature	0.857
gfs_temperature_92500	0.852
gfs_temperature_90000	0.842
gfs_temperature_85000	0.824
gfs_temperature_80000	0.808
cmc_0_0_6_2	0.807

Table 1: Features most correlated to target value in the training dataset according to Pearson metric.

Feature	Wasserstein distance
wrf_available	**49.000**
gfs_available	**49.000**
cmc_available	**49.000**
gfs_2m_dewpoint_grad	24.006
cmc_0_1_0_0	17.057
gfs_precipitable_water	16.999
gfs_v_wind	16.866
gfs_temperature_15000	14.541
gfs_total_clouds_cover_high	13.718
fact_latitude	13.625

Table 2: Most shifted features between train and test datasets according to Wasserstein metric.

3 Pre-Processing

Different models call for different preprocessings and feature selections. We try different approaches in conjunction with one or more models to find the best one in each situation.

- **Dealing with missing values:** either fill missing values with their respective column mean or drop samples containing at least one NaN.
- **Outlier removal:** either remove all datapoints outside of the interquartile range or use sklearn's *LocalOutlierFactor* to detect and exclude outliers.
- **Rescaling:** normalize all features using sklearn's *StandardScaler*.
- **Feature selection based on Random Forest feature importance:** to address the correlation issues mentioned earlier, we keep only the 20 most important features according to a Random Forest Regressor[5]. Interestingly, the first four features represent almost the totality of importance (Tab. 3).

[4] The temperature measurements are taken in close isobaric levels

[5] *n_estimators*=10, *max_depth*=30

- **Feature selection based on Wasserstein distance:** rank features according to the metric described in Subsection 2.3 and keep only the n less-shifted ones as they should better represent the test distribution.
- **PCA:** correlation analysis shows many features are correlated, this poses a problem for linear models. We address this issue by performing a PCA keeping either *num_features* - 1 principal components or by selecting the components explaining 99% of variance.
- **Daylight and hour:** use *fact_time, fact_latitude, fact_longitude* to compute the hour of day and the amount of daylight received in the day the measurement took place.

Feature	Importances
wrf_t2_interpolated	**0.871038**
gfs_temperature_97500	0.034633
cmc_0_0_6_2	0.011729
climate_temperature	0.010516
wrf_snow	0.004567
sun_elevation	0.004368
cmc_0_0_7_2	0.004134
cmc_0_1_0_0	0.003386
gfs_soil_temperature	0.002967
fact_longitude	0.002202

Table 3: The 10 most important features extracted with a Random Forest with 10 trees and max depth equal to 30

4 Models

We perform an initial exploratory analysis comparing a variety of models along with different preprocessings and pick the ones that perform best to be further explored with hyper-parameter tuning.

We start our analysis with a simple explainable model, **Ridge**. The main challenge in its usage is to remove the collinearity present in the dataset. To do this we use techniques such as Principal Component Analysis and feature selection with a Random Forest.

Since it is possible that the relationship between target and features is non-linear, we move to a **Random Forest Regressor**. It proves very effective even for a small number of estimators. However, the training time of the model rapidly increases with forest size.

In order to get even better, we move to boosting techniques. Among the several models available, we use **CatBoost** which is a decision tree boosting algorithm. CatBoost builds a set of decision trees consecutively, each one with reduced loss compared to the previous ones. This procedure is repeated for the specified number of iterations.

5 Hyper-parameter tuning

We perform for each model a simple Grid Search on a small amount of hyperparameters with a five-fold cross validation. By comparing the train and validation scores on the best split we select the best performing model and its respective best set of hyperparameters.

To avoid data leakage between train and validation sets we pack both preprocessing and model in a Pipeline[6] that is repeated on each iteration of the five-fold. c

[6] sklearn.pipeline.Pipeline

Pipeline & Model	Hyperparameters Explored	Best	Validation RMSE
StandardScaler, PCA, Ridge	alpha: [0.1, 0.01, 0.001]	alpha: 0.1	2.4
StandardScaler, 20 RF features Ridge	alpha: [0.1, 0.01, 0.001]	alpha: 0.1	2.71
StandardScaler, 20 RF features Random Forest	n_estimators: [2, 5, 8, 10] max_depth: [15, 30, 60]	n_estimators: 10 max_depth: 30	2.76
StandardScaler, CatBoostReg	lr: [0.1, 0.01], depth: [8,10] iterations: [1000, 10000]	lr: 0.1, depth: 10 iterations:10000	1.52

Table 4: Models and hyperparameters tested

6 Results

Applying different combinations of preprocessing techniques, we observe the following outcomes:

- Filling missing values with the column mean or dropping the sample altogether produces comparable results. We choose to always fill with the mean to avoid dropping potentially useful samples;
- Regardless of the outliers removal technique we tried over distinct experiments, we obtain our best results by keeping the outliers in the analysis;
- Rescaling proves always useful in improving performances;
- Despite our efforts in coming up with sensible metrics for feature selection, we find that keeping all features and letting the model choose the most important ones during training is still the best-performing approach also in the presence of spatial shift;
- PCA is needed only for linear models to avoid collinearity. This leads to noticeable improvements on the train and test scores at the cost of explainability, as PCA changes the basis in which data is represented. Other models show a reduction in performance when PCA is applied;
- Daylight and hour do not lead to any noticeable improvements on any analyzed model. This is possibly due to the feature *sun_elevation* representing similar informations.

Regarding the models, we obtain increasingly better results as the complexity of the model increases. Indeed, the best performing model is the CatBoost regressor with a validation score of 1.52. Despite the results, dealing with more complex models has many drawbacks such as the loss of explainability or the increasing time complexity.

These results suggest that this dataset hardly requires any preprocessing or feature selection. The highest scoring methods either perform regularization or implicit feature selection thanks to trees, showing that an automatic choice of features wins over a curated manual one. This does not surprise us: plenty of features correlate extremely with the target and require no fancy transformation to be used effectively by the models.

7 Conclusions

We show a simple way in which Catboost can be used to predict temperature measurement with very little preprocessing required. This solution yields very good results despite its simplicity.

Despite consisting of almost 2 million measurements, the actual weather stations are few in number, totaling around 5000. A more detailed analysis could investigate aggregates of measurements coming from the same station in order to increase performance. An even more refined approach would be to group together measurements coming from the same station and treat the prediction as a time series problem, applying models such as RNNs and LSTM.