applied sciences

Article

Prediction of Air Pollution Concentration Based on Backward Elimination and Regression Models

Aleksandar Trenchevski [1]

[1] Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University, Skopje, Republic of North Macedonia

* Correspondence: atrenchevski@gmail.com (A.T.); Tel.: +38970363910

Received: date; Accepted: date; Published: date

Abstract: Air pollution has become a global environmental problem, in both developed and developing countries. It has greatly impacted the health and lives of millions of people with increased mortality rates and a lot of illness reports. This article's main goal is to provide machine learning methods of predicting the rates of increased pollution in an area by processing the gathered data from multiple stations, generating features for more precise prediction and selecting the best parameters for tuning a model. The data provided in this article was provided by the repository of AirCare which held pollution information from multiple stations and the DarkSky API for weather data from different cities which included the temperature, humidity and wind characteristics. The development process was consisted of manual feature generation for adding features to increase the accuracy, feature selection for removing redundant features with low correlation with the target variable, training and optimizing multiple regression models with parameters suitable for best precision. All of the pollutants: CO, NO2, O3, PM2.5, PM10 as well as the air quality index (AQI) were used as a target variable during the development of this article.

Keywords: air pollution; feature selection; machine learning; prediction; regression models

1. Introduction

With the increase in computing power and the development of machine learning methods, it opens up the possibility of a large amount of data on atmospheric characteristics collected over a number of previous years to be viewed in a new way and integrated into a common picture. These new methods make it possible to detect the interconnections within the data and to present these results effectively.

Increasing demand for energy, population growth, economic development, urbanization and transportation, the problem of air pollution is the focus of modern society, primarily because of its adverse effects on human health, the environment and the climate system. From the beginning of the Industrial Revolution to the present day, the concentration of harmful substances in our atmosphere has been constantly increasing, but this problem has only recently been approached with greater care. The main pollutants in the air are carbon oxides (carbon monoxide and carbon dioxide), oxides of nitrogen and sulfur, particulate matter (PM2.5 and PM10), ammonia, some toxic metals, volatile organic compounds, etc.

In addition to the air monitoring comes the prediction of increased pollution rates in a particular time which can aid the people in their movement as well as organizations responsible for maintaining the traffic control and industrial factories which are main polluters and source of the toxic materials present in the air.

This article focuses on predicting the hourly air pollution for multiple stations across the country in 2018 from a dataset made from gathered data from the past 4 years (2015 - 2018). Multiple regression models were used for benchmarking and pinpointing the most precise model.

2. Data preparation

The data used in this research consisted from the publicly available git repository from AirCare which had archived data for 4 years (2015 – 2018) and weather data gathered from the open API provided by DarkSky for cities and municipalities across the world, in this case from Republic of North Macedonia. I have combined the data into multiple reports which had the merged info from pollutants as well as the atmospheric details of the local weather.

A total of 10 stations across the country were included in this article's research and all of them had different measurement inconsistencies and had a huge time gap which I had to deal with when merging the data from the datasets.

All of the available pollutants from the dataset were included in each of the consecutive steps: feature generation, feature selection, training and prediction phases because they all contribute to the pollution rates and overall air quality.

Before the training and prediction phases can begin, I had to filter out the redundant and unnecessary data in the combined reports, because there were variables that had very little amount of data values depending on the station that monitored those values and the best decision was to eliminate them. The next step was to fill out missing values which had very little time difference so the manually filled out data with methods for interpolation does not change the realistic values drastically and cause problems during the training phase. Some of the variables like the air quality index (AQI) can be calculated as the maximum index of all pollutants. I added a method for calculating missing AQI values from the other pollutants. The last step included the removal of potential outliers which concluded the dataset to be ready for the next phase.

3. Methodology

a. Feature selection

Feature selection is the process of selecting a subset of relevant features for use in model construction which leads to several benefits:

- Accuracy improvement
- Overfitting risk reduction
- Speed up in training
- Improved Data Visualization
- Increase in explainability of the model

Time series data affecting air pollution contains rich, but also irrelevant and redundant information. This information reduces the accuracy of the predictions and efficiency of the model. There are many feature selection algorithms which are distinguished by the evaluation metric and they are divided into three main categories: filters, wrappers and embedded methods.

In this article the backward elimination method from the wrapper category is used due to its precision for selecting relevant features based on the given machine learning model, but for a large amount of features the time complexity rises.

Backward Stepwise Regression is a stepwise regression approach that begins with a full (saturated) model and at each step gradually eliminates variables from the regression model to find a reduced model that best explains the data. Also known as **Backward Elimination** regression.

The stepwise approach is useful because it reduces the number of predictors, reducing the multicollinearity problem and it is one of the ways to resolve the overfitting.

The subset of features is generated for each station, target variable and machine learning model accordingly for maximum efficiency of the algorithm and better testing results.

b. Regression learning

The selected subset of features is used as an input in the training of 6 regression models for predicting pollution values for the year 2018 which is used as the test dataset for each model.

The execution process starts from the first model and collects all the prediction values, errors from predicting as well as the selected features for each iteration. There is some manual feature generation for adding features derived from the timestamp and the previous value of the target variable, as well as categorical features which were needed to be hot encoded because the regression algorithm cannot process string object features.

Selecting proper parameters for tuning the efficiency of the model is calculated using randomized grid search due to the time complexity of the grid search algorithm for a large number of parameters.

- ***Decision tree regression***

The first regression model is the decision tree. This tree builds regression models in the form of a tree structure. It breaks down the dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. This model was one of the most accurate models ranking mostly at second or third place.

- ***Dummy regression***

The dummy regression model was the most inaccurate model because it calculates the predictions by following a set of simple rules. It can be set to predict a fixed value calculated as the mean, median and quantile of the training set or as a constant given by the user. It was used as a baseline to compare the rest of the models.

- ***Light GBM regression***

The third and the most precise model used during the prediction tests is the Light GBM regression model. It is a fast, distributed, high-performance gradient boosting framework based on the decision tree algorithm, used for ranking, classification, regression and many other machine learning tasks.

It's based on decision tree algorithms; it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf wise. So, when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm and hence results in much better accuracy which can rarely be achieved by any of the existing boosting algorithms. Also, it is surprisingly very fast, hence the word 'Light'.

Leaf wise splits lead to increase in complexity and may lead to overfitting and it can be overcome by specifying another parameter max-depth which specifies the depth to which splitting will occur.

Light GBM possesses multiple advantages on why it is the best choice for dataset training and calculating predictions.

- **Faster training speed and higher efficiency:** Light GBM uses histogram-based algorithm i.e. it buckets continuous feature values into discrete bins which fasten the training procedure.
- **Lower memory usage:** Replaces continuous values to discrete bins which result in lower memory usage.
- **Better accuracy than any other boosting algorithm:** It produces much more complex trees by following leaf wise split approach rather than a level-wise approach which is the main factor in achieving higher accuracy. However, it can sometimes lead to overfitting which can be avoided by setting the max_depth parameter.

- **Compatibility with large datasets:** It is capable of performing equally good with large datasets with a significant reduction in training time as compared to XGBoost.
- **Parallel learning supported**

- ***Linear regression***

Linear regression is a machine learning algorithm based on supervised learning. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

Linear regression performs the task to predict a dependent variable (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y (output).

Training the linear regression model means trying to find out coefficients for the linear function that best describe the input variables.

While building a linear mode the main goal is to minimize the error made by the algorithm while making predictions, which is done by choosing a function to help measure the error also called a cost function. The cost function of the linear regression is the **Root Mean Squared Error (RMSE)** between the predicted y value and the true y value.

Estimating the coefficients to reduce the cost function is done by using the mathematical algorithm called **Gradient Descent** achieving the best fit line. The idea is to start with random values for the coefficients and then iteratively updating the values, reaching the minimum cost.

- ***Random Forest regression***

Random Forest is a flexible, easy to use machine learning algorithm that produces great results most of the time with minimum time spent on hyper-parameter tuning. It has gained popularity due to its simplicity and the fact that it can be used for a great amount of regression tasks.

Random Forest is an ensemble machine learning technique capable of performing regression tasks using multiple decision trees and a statistical technique called bagging.

A Random Forest regression instead of just averaging the prediction of trees it uses two key concepts:

- Random sampling of training observations when building trees
- Random subsets of features for splitting nodes

Random Forest builds multiple decision trees and merges their predictions together to get a more accurate and stable prediction rather than relying on individual decision trees.

There are several advantages that characterize the efficiency of this model:

- Reduction in overfitting: by averaging several trees, there is a significantly lower risk of overfitting
- It is very easy to measure the relative importance of each feature on the prediction
- Small number of hyperparameters and the default settings often produce a good prediction result
- It has an in-built validation mechanism named Out-of-bag

However, using the Random Forest model also comes with a set of disadvantages that may require to swap the model for a more suitable regressor:

- It’s more complex and computationally expensive than the decision tree algorithm. This makes the algorithm slow and ineffective for real-time predictions as a more accurate prediction requires more trees
- Cannot extrapolate at all to data that is outside the range that the algorithm has seen

Random Forest as a technique is consisted of many simple ideas combined together to yield an extremely accurate model.

- ***Support Vector regression***

Support Vector regression (SVR) is characterized by the user of kernels, sparse solution and VC control of the margin and the number of support vectors. Although less popular than Support Vector Machine (SVM), SVR has been proven to be an effective tool in real-value function estimation. As a supervised-learning approach, SVR trains using a symmetrical loss function, which equally penalizes high and low misestimates.

One of the main advantages of SVR is that its computational complexity does not depend on the dimensionality of the input space. Additionally, it has excellent generalization capability, with high prediction accuracy.

The regression problem is a generalization of the classification problem, in which the model returns a continuous-valued output, as opposed to an output from a finite set. In other words, a regression model estimates a continuous-valued multivariate function.

SVR formulates this function approximation problem as an optimization problem that attempts to find the narrowest tube centered around the surface, while minimizing the prediction error, that is, the distance between the predicted and desired outputs.

- ***XGBoost regression***

XGBoost is an optimized distributed gradient boosting model designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solves many data science problems in a fast and accurate way.

Gradient Boosting Machines fit into a category of machine learning called Ensemble Learning, which is a branch of machine learning methods that train and predict with many models at once to produce a single superior output.

Ensemble learning is broken up into three primary subsets:

- **Bagging:** Bootstrap Aggregation or Bagging has two distinct features which define its training and prediction. For training it leverages a Bootstrap procedure to separate the training data into different random subsamples, which different iterations of the model use to train on. For prediction, a bagging regression will take an average of all models to produce an output. Bagging is typically applied to high variance models such as Decision Trees and the Random Forest algorithm is a very close variation on bagging.
- **Stacking:** A Stacking model is a "meta-model" which leverages the outputs from a collection of many, typically significantly different, models as input features. For instance, this allows you to train a K-NN, Linear Regression and Decision Tree with all of your training data, then take those outputs and merge them with a Logistical Regression. The idea is that this can reduce overfitting and improve accuracy.
- **Boosting:** The core definition of boosting is a method that converts weak learners to strong learners and is typically applied to trees. More explicitly, a boosting algorithm adds iterations of the model sequentially, adjusting the weights of the weak-learners along the way. This reduces bias from the model and typically improves accuracy.

Bagging along with boosting are two of the most popular ensemble techniques which aim to tackle high variance and high bias.

In conclusion, the XGBoost algorithm is optimized for modern data science problems and tools. It leverages the techniques mentioned with boosting and some of the major benefits of XGBoost are that its highly scalable/parallelizable, quick to execute and typically out performs other algorithms.

4. Experimental results

The experimental results of the used algorithms were made possible by dividing 75% of the dataset for training which include the first 3 years (2015 – 2017) and 25% for testing which is the target year 2018. A cross-validation algorithm known as **Randomized Search** was used to determine the maximum potential of each algorithm.

The evaluation metrics used for ranking each of the results provided was **Mean Absolute Error** (MAE) as one of the most often used metrics with regression models. In MAE the error is calculated as an average of absolute differences between the target values and the predictions. The MAE is a linear score which means that all the individual differences are weighted equally in the average.

I have posted the error and prediction plots for the PM10 pollutant derived from each of the algorithms used because it had the most consistent data for training and testing.

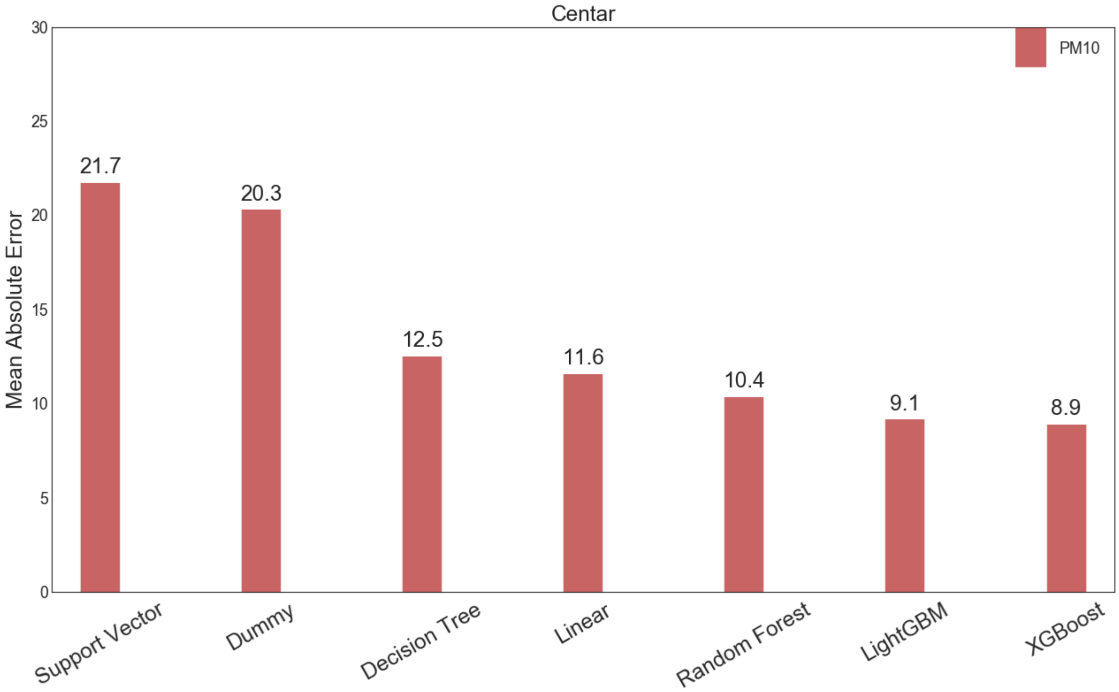

Figure 1. Mean Absolute Error plots for each algorithm predicting the PM10 pollutant in the center of Skopje

From the figure we can conclude that the XGBoost is the most precise algorithm for predicting the PM10 pollutant as a target variable with a MAE value of 8.9, right after comes Light GBM as the second-best algorithm with a mean absolute error of 9.1 and Random Forest with 10.4. The overall performance improvement of these 3 models is slightly below 60%.

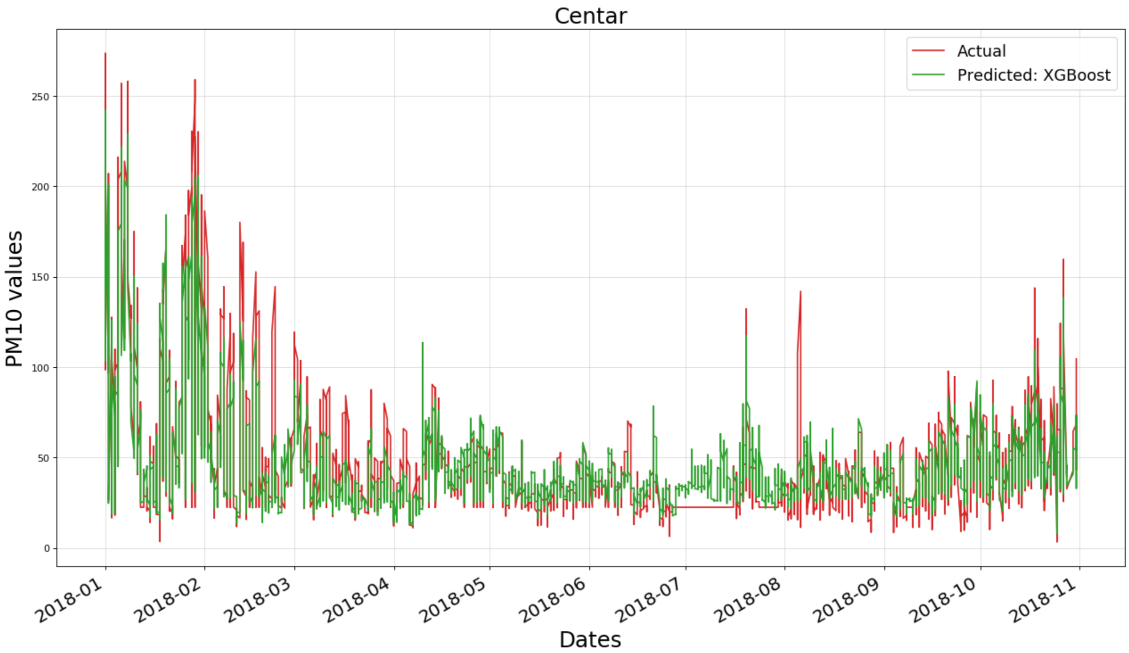

Figure 2. Plot showing the actual and predicted values of the PM10 pollutant for each hour in 2018 in the center of Skopje

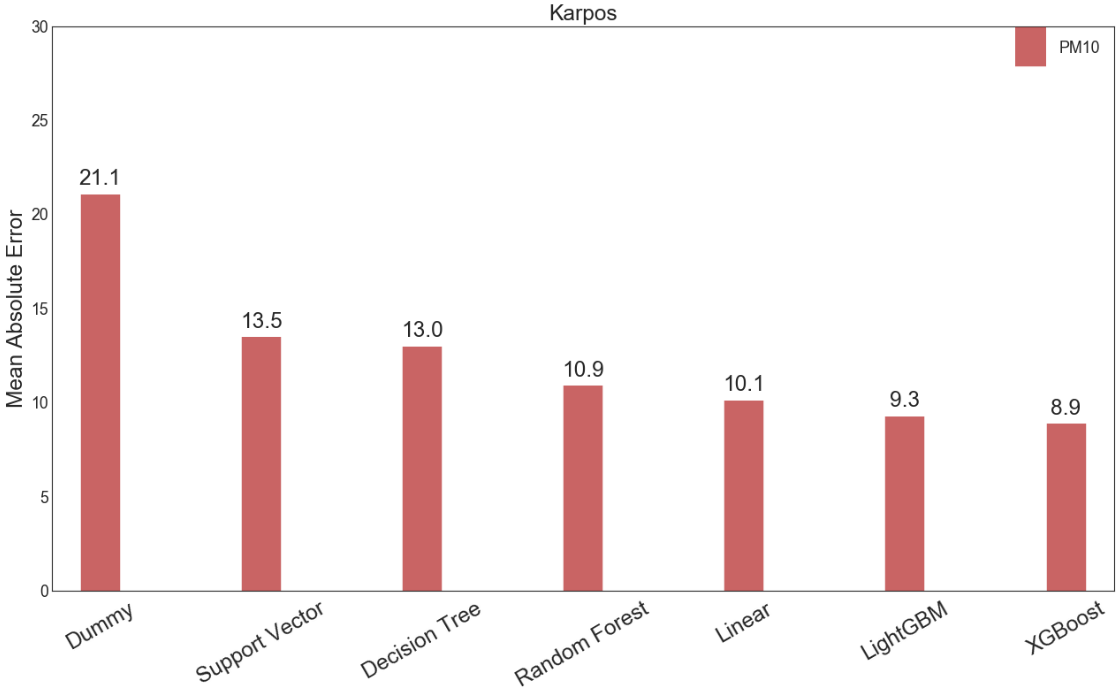

Figure 3. Mean Absolute Error plots for each algorithm predicting the PM10 pollutant in the municipality of Karpos in Skopje

Best regression models for predicting PM10 values in the municipality of Karpos are XGBoost and LightGBM with a MAE value of 8.9 and 9.3 respectively which is a solid performance improvement by around 60%.

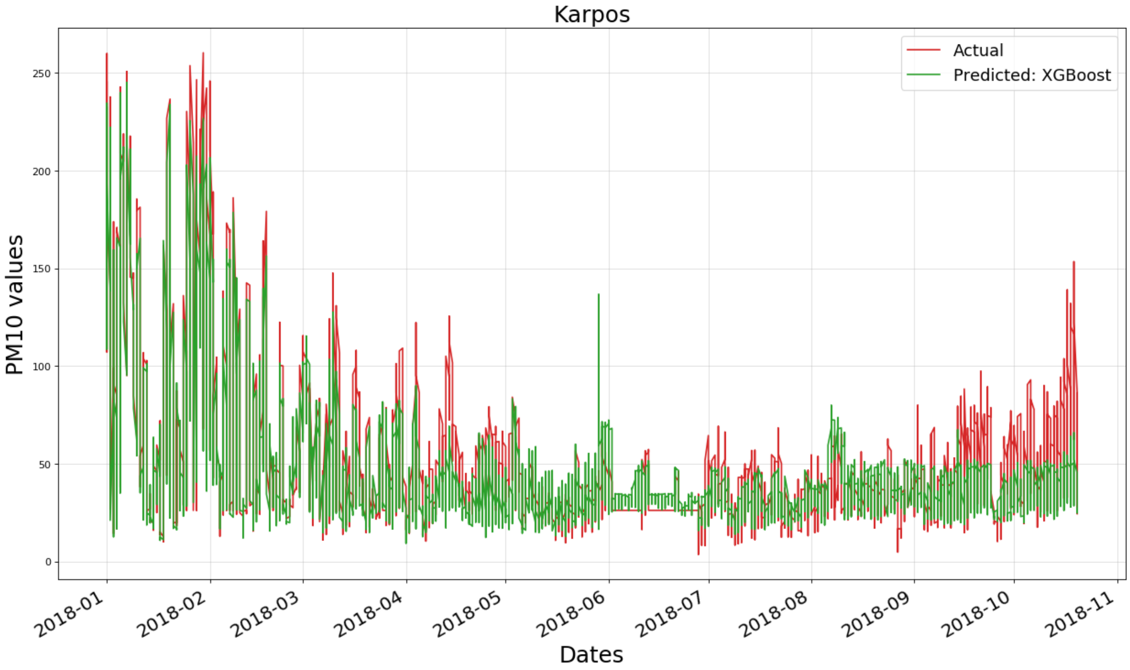

Figure 4. Plot showing the actual and predicted values of the PM10 pollutant for each hour in 2018 in the municipality of Karpos in Skopje

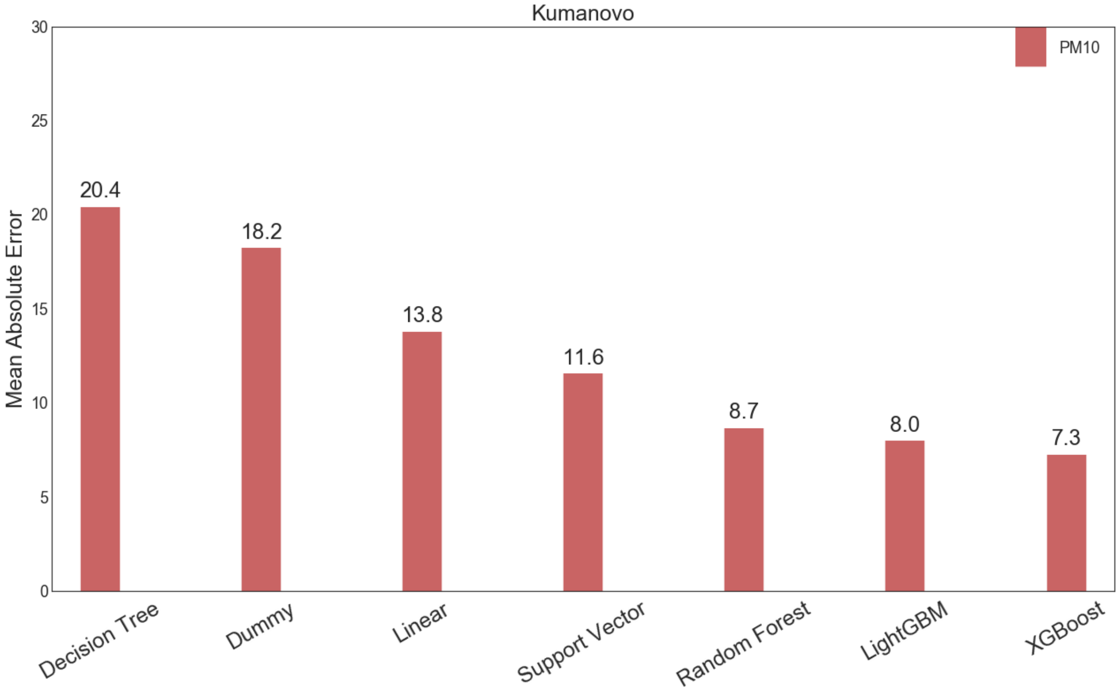

Figure 5. Mean Absolute Error plots for each algorithm predicting the PM10 pollutant in Kumanovo

The 2 best regression models so far are XGBoost and Light GBM which have the best mean absolute errors, 7.3 and 8.0, when predicting the PM10 values for data gathered from the pollution sensors in Kumanovo which results in a 60% performance improvement over the baseline algorithm.

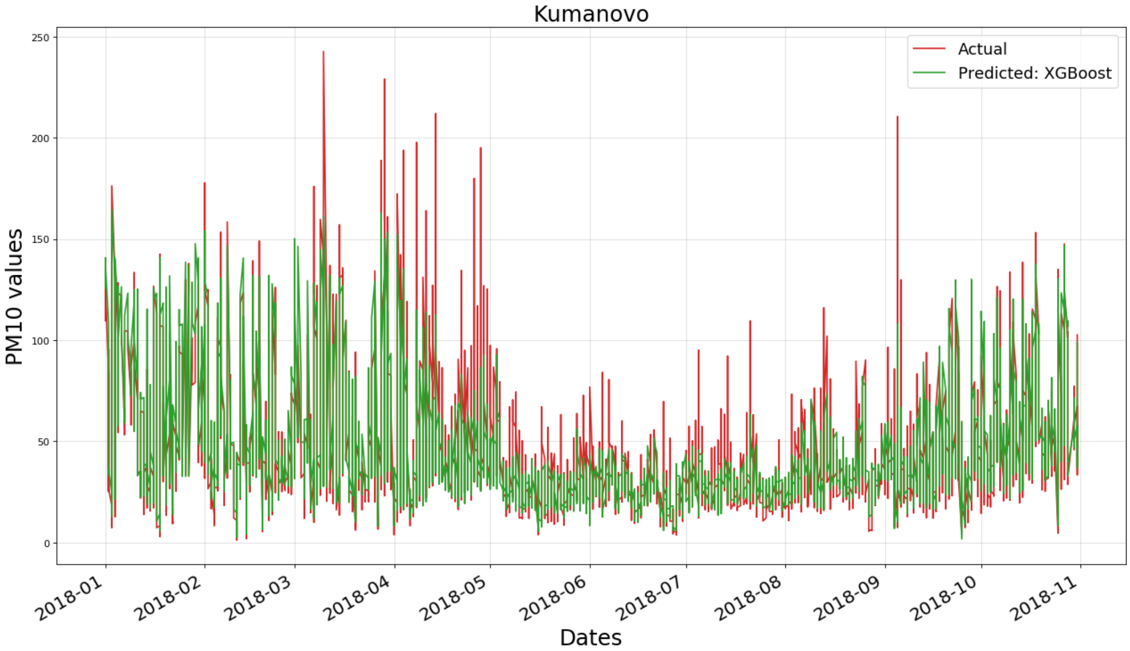

Figure 6. Plot showing the actual and predicted values of the PM10 pollutant for each hour in 2018 in Kumanovo

5. Conclusions

This study aims to provide a machine learning concept and provide an approach on the necessary steps that need to be taken into account when working with datasets with realistic data.

The first step is preparing and filtering the used dataset for training and testing to eliminate unnecessary features, remove outliers that corrupt the data and remove any inconsistencies with the target variables in order to preserve data integrity. Next comes manually generating useful features from already existing features in the dataset, use popular feature selection algorithms to improve overall dataset accuracy, use different cross-validation algorithms to achieve best results and obtain useful hyperparameters for each regression model. The final step is the usage of evaluation metrics for dealing with prediction results from multiple regression models in order to obtain the best result and highlight it on a table for direct comparison.

The overall results manage to perform better than the baseline algorithm (Dummy Regression) by 60% and deliver a low mean absolute error which confirms the necessity of each step mentioned above for providing best results.

The incomplete data played a major role in making the whole process harder to develop due to its inconsistency, a lot of outliers, missing values that needed to be filled by interpolation or removed entirely. From around 30000 – 40000 rows of data it had to be cut down to around 15000 – 20000 which significantly lowers the accuracy of the model when training with half of the entire data.

References

1. Backward Stepwise Regression. Available online: http://www.analystsoft.com/en/products/statplus/content/help/analysis_regression_backward_stepwise_elimination_regression_model.html (28.12.2019).
2. Decision Trees in Python with Scikit-Learn. Available online: https://stackabuse.com/decision-trees-in-python-with-scikit-learn/ (28.12.2019).
3. Linear Regression using Python. Available online: https://medium.com/analytics-vidhya/linear-regression-using-python-ce21aa90ade6? (28.12.2019).
4. Random Forest Regression model explained in depth. Available online: https://gdcoder.com/random-forest-regressor-explained-in-depth/ (28.12.2019).
5. Support Vector Regression Or SVR. Available online: https://medium.com/coinmonks/support-vector-regression-or-svr-8eb3acf6d0ff (28.12.2019).

6. A Step by Step Regression Tree Example. Available online: https://sefiks.com/2018/08/28/a-step-by-step-regression-decision-tree-example/ (28.12.2019).
7. Awad M., Khanna R. (2015) Support Vector Regression. In: Efficient Learning Machines. Apress, Berkeley, CA
8. Tuysuzoglu, G.; Birant, D.; Pala, A. Majority Voting Based Multi-Task Clustering of Air Quality Monitoring Network in Turkey. *Appl. Sci.* **2019**, 9, 1610.
9. Xu, X.; Ren, W. Prediction of Air Pollution Concentration Based on mRMR and Echo State Network. *Appl. Sci.* **2019**, 9, 1811.