USE-CASE REPORT

Runoff Prediction using Machine Learning Algorithms

OBJECTIVE

The objective of this use-case is to predict the runoff from the rainfall parameters using Machine Learning Algorithms

Abstract

Quality of Runoff is a matter of concern because it interferes in health and environmental aspects, drainage, land use, irrigation and power generation. Runoff occurs when the intensity of rainfall exceeds the infiltration rate at the ground surface. So, when the soil is saturated then there is increase in both downward movement of water and lateral movement in form of seepage flow. The rate of runoff is required for the design of drains, canals and other channels, and for the prediction of water levels in streams and rivers. Quantity of runoff is required when storage is involved for irrigation, power generation, river transport etc. The conventional method of runoff measurements include use of current meters and and rated cross-section channels for flow of water. Machine Learning methods can be applied here to predict runoff from the rainfall and weather conditions.

Contents

Introduction

What is Machine Learning?
Application of AI which provides systems an ability to automatically learn and improve from experience without being explicitly programmed.

Accurate runoff forecasting will help us in effective planning and use of water resources. So, using the rainfall and runoff parameters we try to create a machine learning model which can forecast runoff.

Dataset and Description

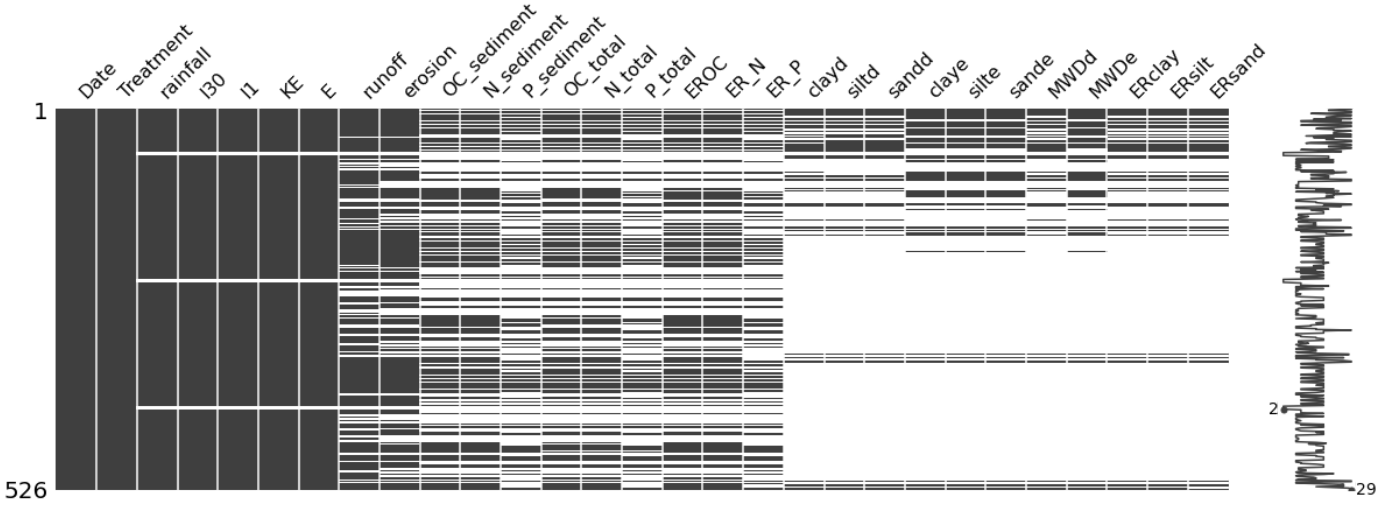
For this use-case, we used the data derived from the Europan Commission H2020. Different weather and soil parameters were measured for a duration of 6 years from 2010 to 2015. The sampling location were different agricultural lands.

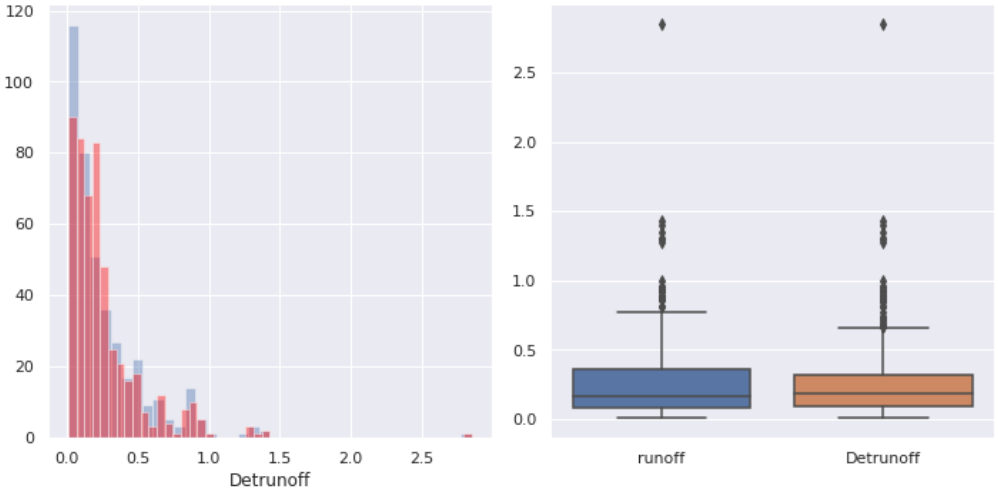
The dataset consisted of 526 instances of 18 features. The features were the measurement of mm of rainfall, intensity of rainfall for 30 min and 1 hr, rainfall erosivity index, kinetic energy, runoff, erosion, and seveal other chemical and soil contents.

Data Pre-processing

The dataset needed cleaning and modifications. The categorical data were converted to numerical data. Many columns had more than 50% of the data missing so it was necessary to remove them. Data Pre-processing is necessary because features having null values make the model in-consistent which results in an inaccurate model. Also, categorical Data cannot be understood by the computer so it is necessary to convert it to numerical form.

Various data imputation techniques like considered like imputation from mean, median, random imputation and imputation using Linear Regression. The white spaces in the bars represent missing values.





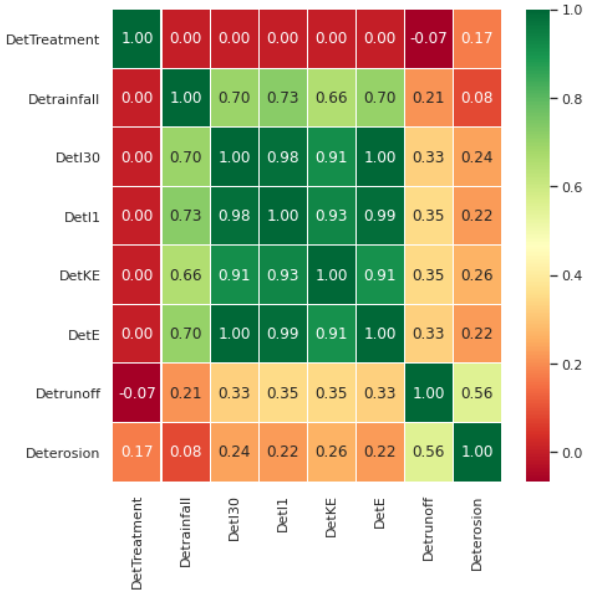
Mean, Median imputation only look at the distribution of the values of the variable with missing entries. But since there is correlation between the missing values and other variables so to get more accurate values regression methods are used. Above plots show result after random imputation and imputation using Linear regression,

Feature Selection

In this process we we selected those features which contribute most to the target variable and remove those features which affects negatively to the model performance.

These features do not contribute to model performance and they also increase the computation time. Feature Selection is an important process because it helps in improving the accuracy of the model, reduces overfitting because the model is not being trained on noisy data, and also training on fewer features requires less time.

Some methods which were used for finding important features are calculation of standard deviation, correlation values and using library functions.



Training and Testing data

In this step the data is transformed and divided into training and testing set.
The transformation method used is min-max normalization which transforms every feature in a range of zero to one.
The train and test ratio is 85 : 15.

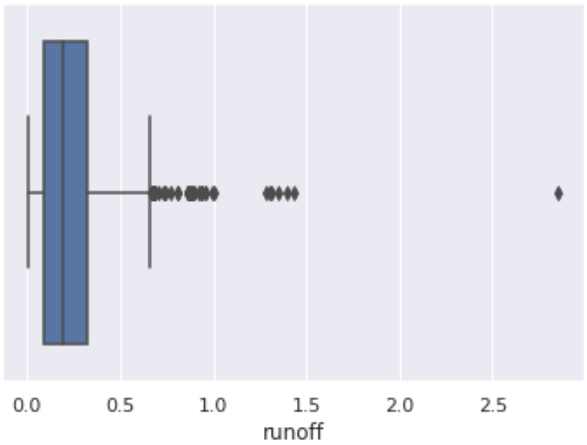
Cross Validation and Hyper Parameter Tuning

We perform Cross Validation and Hyper Parameter tuning for every learning model.
We test the model on unseen data to see if the model overfits or underfits. Cross validation helps to train and test the model on entire data by splitting the data in k folds. It helps us to test the effectiveness of our machine learning model. By finding the mean of all K folds we get the exact error produced by our model.

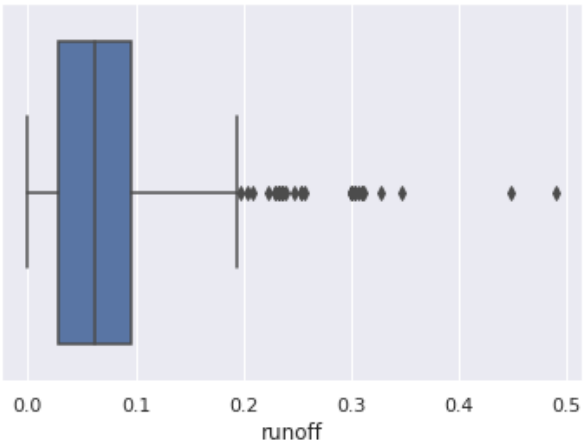
Hyperparameter tuning is a method to choose the best parameters for any model for which the performance of the model is best. The parameters like max iterations, max depth, learning rate, criterion etc. govern the training process of any model and produce a significant impact on the accuracy and error produced by the model. We used Grid search method for determination of best parameter for every model.

Outlier Handling

The presence of outliers in a data set results in a poor fit and lower predictive modeling performance. The data used by us contains lot of outliers so we use Isolation Forest algorithm to remove some outliers. Isolation Forest has different hyper parameters which can be used for outlier handling. The most important parameter "contamination" is kept 0.1 in our case to remove only highly skewed values since we have very less amount of data.



Before outlier Removal



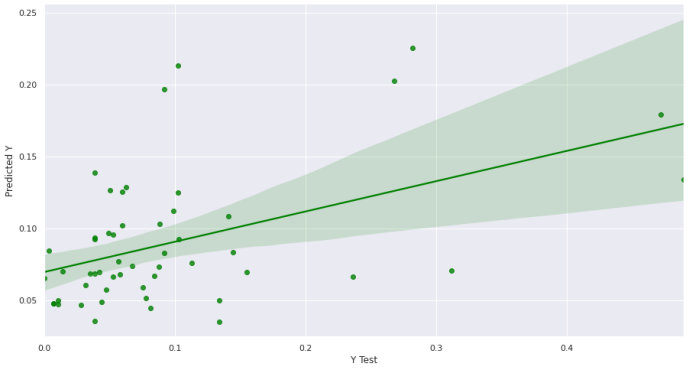
After outlier Removal

Learning Models

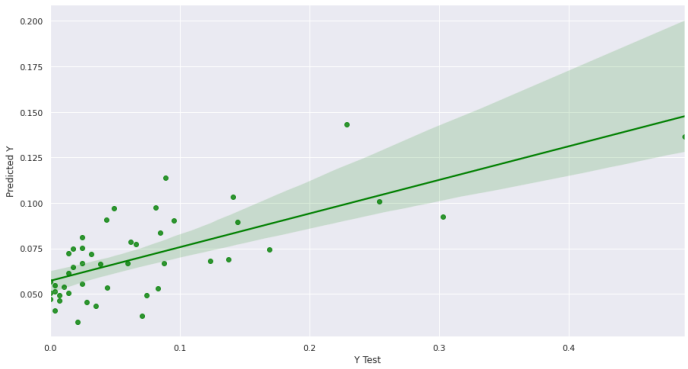
We trained and tested our dataset on different machine learning and deep learning models. Some Learning models like Ridge Regression, Decision tree regression, Support Vector regression and Random Forest regression were used for prediction of runoff values.

Ridge Regression

Ridge Regression easily handles the problem of multi collinearity in linear regression which mainly occurs in high dimensional data. Ridge regression uses linear least squares loss function with l2 regularization.



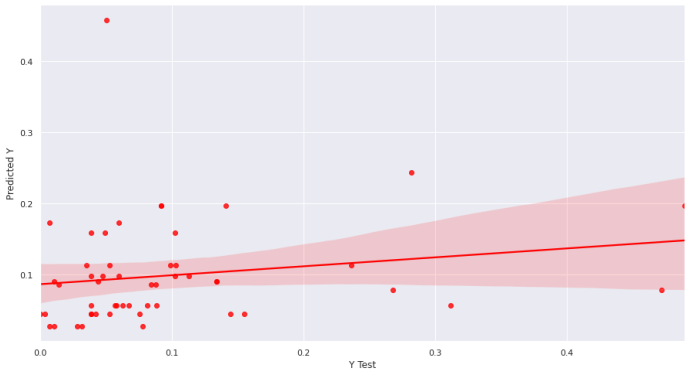
Error values with outliers
MAE: 0.059
MSE: 0.0081
RMSE: 0.0904



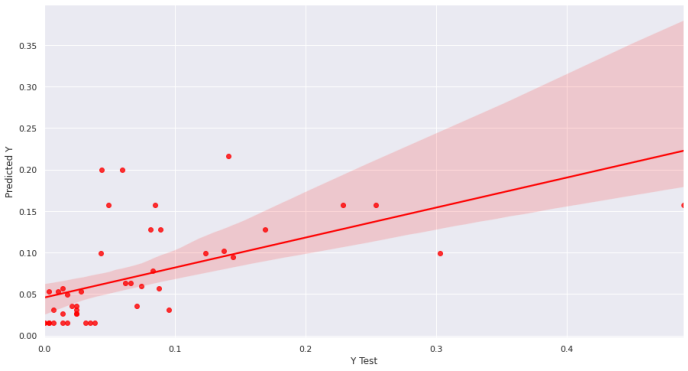
Error values after removing outliers
MAE: 0.051
MSE: 0.0059
RMSE: 0.077

Decision Tree Regression

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a 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.The hierarchy of the tree is decided on the criterion function which calculates errors to split the tree.



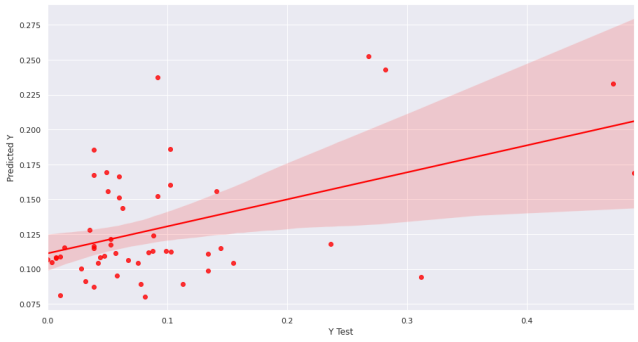
Error values with outliers
MAE: 0.0708
MSE: 0.0132
RMSE: 0.115



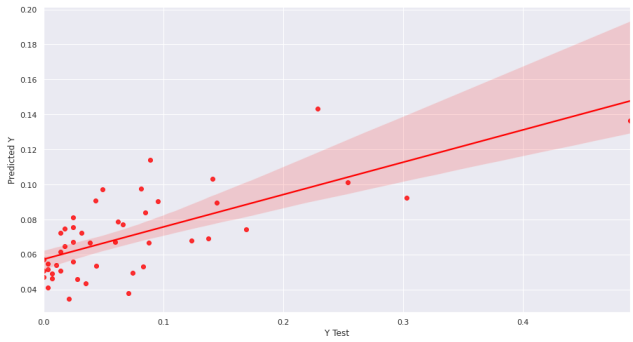
Error values after removing outliers
MAE: 0.045
MSE: 0.0057
RMSE: 0.075

Support Vector Regressor

SVR gives us the flexibility to define how much error is acceptable in our model and will find an appropriate line (or hyperplane in higher dimensions) to fit the data. However support vector regressor does not perform good in case of high dimensional data.



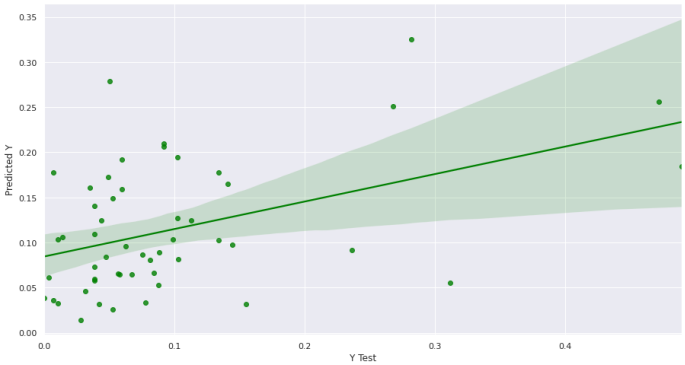
Error values with outliers
MAE: 0.076
MSE: 0.0093
RMSE: 0.096



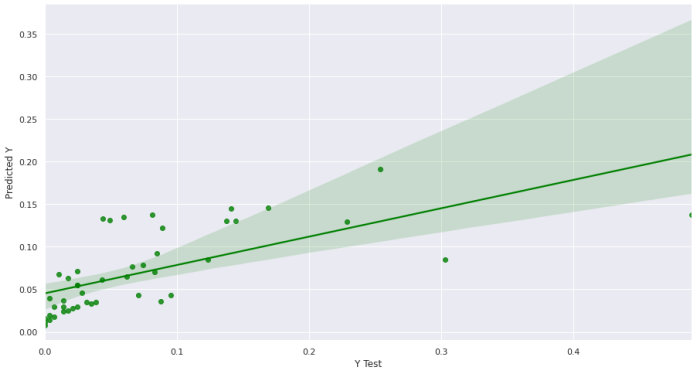
Error values after removing outliers
MAE: 0.05
MSE: 0.0059
RMSE: 0.077

Random Forest Regressor

Random forest is a Supervised Learning algorithm which uses ensemble learning method for both classification and regression. It is also called a s meta-estimator as it combines different decision trees. The number of decision tree can be controlled by n_estimator parameter which also helps to get better model. Each tree samples from the original sample using different randomness which prevents the model from overfitting. In most cases random forest acts like an improvement over decision tree but it requires high amount of compilation time than decision tree regressor.



Error values with outliers
MAE: 0.068
MSE: 0.0094
RMSE: 0.0971



Error values after removing outliers
MAE: 0.038
MSE: 0.0050
RMSE: 0.0709

Technology and Framework

- I used python language it is a powerful high-level language and I am well versed with it.
- I used scikit learn library for the development of Machine Learning models as scikit learn provides many useful libraries and has very good documentation of its tools.
- For the data handling, computation, and visualization purposes Pandas, numpy, seaborn, and matplotlib was used.
- I used Google colaboratory as our platform so that we can use it's fast and free TPU accelerators which decreases model training and testing time to a great extent.

Conclusion

From the above use-case, we determined the important features which affect runoff. The models were trained on a insufficient amount of data and so the results can be improved using advanced algorithm like LSTM when trained with high amount of data. Since the runoff gets affected by the factor of time and LSTM algorithm will help us to forecast runoff values for distant values of time. Also training against large dataset will help us to further reduce error and predict more accurate results.