Optimizing Algorithms for Predicting Soil Organic Carbon Concentration

EECS 498-010 Final Project Report
Stefanos Agamemnon Frilingos - EE, BSE

1 Introduction

Monitoring the level of soil organic carbon (SOC) is an integral part in evaluating an area's ground quality, since it is directly related to the biological and chemical properties of soil. SOC is also an indicator of the vegetation growth that captures carbon and assists in the fight against climate change. In-person sampling has been the traditional way of measuring such soil characteristics as SOC, by inserting long tubes into the ground and performing thorough chemical analysis. However, with the popularization of Artificial Intelligence (AI) and Machine Learning (ML) methods, new research has exposed innovative and efficient methods in measuring soil characteristics. Remote sensing, using satellites that can produce images, or radar/lidar point clouds are essential in capturing a given field study area. By using different frequency bands, scientists can calculate vegetation indices as well as build a Digital Elevation Model (DEM) which produces a lot of important topographic features. All these features and indices can assist in training models to monitor and predict the level of SOC in new, unsampled areas. Our project aims at concentrating on a specific study area, and optimizing six different algorithms to predict the SOC levels. The purpose is to research how different ML algorithms perform in the prediction of such a target variable, and work on improving the accuracy in order to lead the way towards a well enough performing program that can be trusted to monitor SOC on a global scale.

2 Motivation

As the charge against climate change becomes more vital by the day, new technology to help this fight is essential. Taking in-person samples to monitor SOC is far too expensive and time consuming. AI technologies that use remote sensing combined with DEM data is a more efficient way to estimate soil properties such as SOC, which is used as a climate change indicator. Due to the very limited research and datasets available on such topics of modeling soil properties through remote sensing, we were motivated to contribute to the scientific community by building upon a publicly available dataset and investigating best methods for predicting SOC concentration on land. By optimizing and adapting such an algorithm on a global scale, it is possible to eradicate the need for in-person soil sampling, and assist in accurately keeping track of our environment in real-time.

3 Preliminary

The study area was chosen based on the research limitations of similar papers available online. The paper published by *Laamrani et al. 2022*, is the only publicly available dataset with sufficient samples of SOC levels, DEM model of topographic derivatives, and vegetation indices which are calculated using satellite imagery. Other research papers on similar topics (*Zhou et al. 2020, Goetz et al. 2019*) produced their own dataset but without making it publicly available,

hence *Laamrani et al. 2022* was the only option. The study site is located in southwestern Ontario, Canada in the Lake Erie basin. Figure 1 below depicts three scaled maps of the precise location of the area, and shows the site with color coded elevation on the right plot.

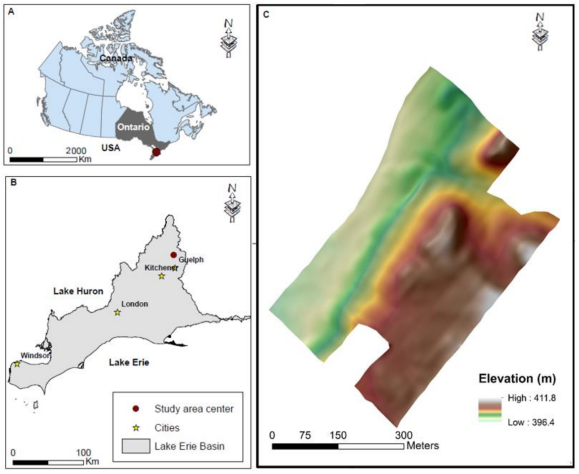

Figure 1: *(A) Location of study area in the state of Ontario, Canada. (B) Red dot depicts study area center, on Lake Erie Basin. (C) Area with elevation.*

The great variability in elevation which spans a range from 396.4 up to 411.8 meters, is crucial for the training of the model with diverse topographic characteristics, which are predicted to increase the correlation between features and the target variable.

The dataset has a collection of 840 soil samples that produced the SOC measurements, along with a total of 54 topographical derivatives. You can find the full dataset on our Gitlab at: (https://gitlab.eecs.umich.edu/frilis/soc-mod). Each sample was spaced approximately 20m apart across the field, as is shown in Figure 2. This figure also depicts the approximate concentration of SOC in different colors, shown by the legend, and by comparing with the color coded elevation model in figure 1C, it is quite apparent that the elevation may be very highly correlated to the SOC levels.

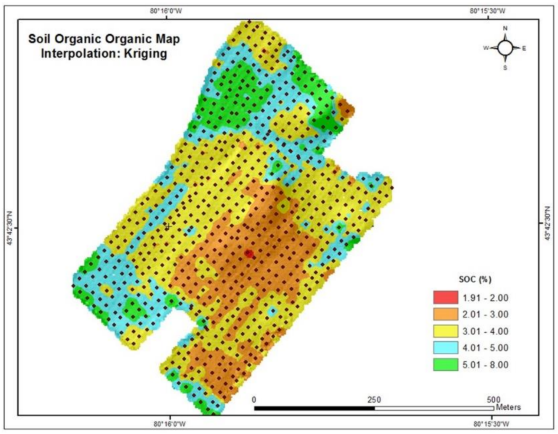

Figure 2: *Map of study area showing each sample, generated at 20-m res through kriging interpolation*

The dataset was passed into our program in its entirety for further interpretation and analysis, as will be discussed in the following section.

4 Methods

The coding work was split into four main sections which consisted of the typical workflow of a ML project: i) Loading in the data and selecting training features ii) Validation testing iii) Testing algorithms iv) Evaluating performance. The scope of this project was not to build our own versions of popular algorithms and fit them into predicting SOC, but rather use popular ML library sci-kit learn, which allows great flexibility in optimizing algorithms for various parameters.

The algorithms were chosen based on previous research on the modeling of SOC, on different datasets, and adopting algorithms that seemed to have performed relatively well (*Zhou et al. 2020)*. The final algorithms selected were: Random Forest Regression (RF), Support Vector Regression (SVR), Stochastic Gradient Descent Regression (SGD), Gradient Boosting Regression (GB), adaBoost Regression (adaB) and Extra Tree regressor (ET). ET is very similar to the RF, in that it creates many decision

trees, but the sampling of each tree is random, without replacement (*Geurts et al. 2006*)

4.1 Feature Extraction

First, we created a function that would select the most important features correlating to SOC levels, since the dataset offered 54 different features, it was inevitable that some would hinder performance. A Random Forest algorithm performs implicit feature selection because it splits nodes on the most important variables. Thus, random forest regression was used to fit the raw dataset with all features, and followingly, the attribute .feature_importances_ offered by the scikit-learn package determined feature importance. All 16 features that showed an importance greater than 0.02 were included in the training. Following the feature extraction, a function was called to split the dataset into 80% training and 20% testing. A depiction of the feature importance is shown as a bar graph in figure 3.

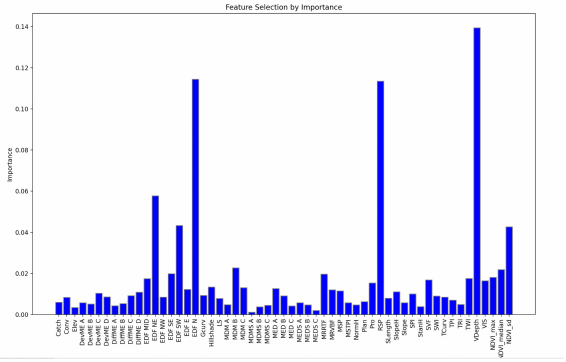

Figure 3: *Bar graph of all 54 feature importance on a scale [0, 1], where all 16 features with values >=0.2 were selected*

The bar with the highest importance value was valley depth (Vdepth) which is a derivative of elevation and an interpolated ridge level, that verifies the significance of elevation in correlation to SOC.

4.2 Validation Testing

Each algorithm was validating for various different parameters. For RF, GB, adaB and ET, following past research and published articles (*Koehrsen, 2018*), as well as analyzing information on the available parameters on the sci-kit learn website, arrays for each of the most important parameters were created with linearly or logarithmically spaced values to validate. A grid dictionary was created which served as input to the randomized cross-validation (CV) function by sci-kit learn. This randomized form of CV ran 100 iterations of 3-fold CV with random_state always equal to zero. Randomized search was chosen over GridSearch, for time-saving purposes, since the latter tests all possible combinations of parameters, which would become too inefficient for our means, especially when we are testing about five parameters for each of the algorithms. Setting the number of iterations to 100 in RandomizedSearch only runs this many times by randomly selecting a combination of parameters. This method only required a fitting of the feature and target training data, comprising 80% of the dataset. The sci-kit attribute “best_params_” allowed to directly output the best performing parameters following cross-validation.

For the two remaining algorithms, SGD and SVR, an extra validation set was extracted from the training data making training 64% and validation 16%. Since these two regression algorithms required standard scaling, a more traditional way of parameter selection was followed. The validation testing was performed in loops, where for SGD two loops found the optimal penalty and loss parameters, while for SVR the optimal C value and kernel were found.

4.3 Algorithm Testing

Following hyperparameter tuning, the ideal parameters chosen are passed into new test

functions that run each algorithm at 80% training and 20% test data. Following fitting of each tuned model, the mean absolute error (MAE) mean squared error (MSE), mean absolute percentage error (MAPE) and coefficient of determination (R^2-score) are calculated in order to holistically evaluate algorithm performance (*3.3. Metrics and Scoring: Quantifying the Quality of Predictions*).

4.4 Performance Evaluation

As mentioned in section 4.3, for regression analysis, MAE, MSE, MAPE and R^2 are popular methods in determining how well a regression model performs. More specifically, MAE, MSE, MAPE and R^2 are calculated through equations (1), (2), (3), (4) respectively:

$$MAE(y, \widehat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} \left| y_i - \widehat{y_i} \right| \quad (1)$$

$$MSE(y, \widehat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \widehat{y_i})^2 \quad (2)$$

$$MAPE(y, \widehat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} \frac{\left| y_i - \widehat{y_i} \right|}{max(\varepsilon, \left| y_i \right|)} \quad (3)$$

$$R^2 = 1 - \frac{RSS}{TSS} \quad (4)$$

The MAE and MSE are relatively straightforward, where the error is the difference between y_i the true value and $\widehat{y_i}$ the predicted value from regression. In MAPE, it should be noted that the variable ε is automatically set to be arbitrarily small yet strictly positive to avoid undefined results when y is zero. In (4) RSS signifies the sum of squares of residuals while TSS refers to the total sum of squares. These can be expressed as in (5) and (6) below.

$$RSS = \sum_{i=1}^{n} (y_i - \widehat{y_i})^2 \quad (5)$$

$$TSS = \sum_{i=1}^{n} (y_i - \overline{y})^2 \quad (6)$$

In (6) $\overline{y}$ represents the mean of the true values y of the SOC level. MAE is also known as the l1-norm loss and is an objective way of evaluating the general performance of a regression. Similarly, MSE, representing the expected value of the quadratic error, is a very popular method of determining how well a regression model fits. MAPE is also known as the mean absolute percentage deviation and it is sensitive to relative errors, meaning it does not change by global scaling of the target variable. Hence, it may be considered more objective than MAE and MSE, which can potentially show slight bias, depending on the linearity of a model. The R^2 score shows how well unseen samples are likely to be predicted by the algorithm, through the proportion of explained variance. Mathematically, it represents the proportion of variance of y that has been explained by the independent variables in the model.

5 Experimental Results

Following validation testing to perform hyperparameter tuning, the parameters shown in tables 1 through 6 were selected to be optimal. This selection was based on evaluating the mean squared error and the R^2 score, for SGD and SVR, and using sci-kit's built-in best_params_ attribute for the rest using 3-fold CV.

Features	Random Forest
n_estimators	200
Max Samples Split	5
Max Features	sqrt(n_features)
Max Depth	80
Bootstrap	True

Table 1: *Optimal parameters of RF regression with CV*

Features	Support Vector
C	1
Kernel	rbf

Table 2: *Optimal parameters for SV regression*

Features	Stochastic Gradient Descent
Loss	squared error
Penalty	None

Table 3: *Optimal parameters for SGD regression*

Features	Gradient Boosting
n_estimators	357
Loss	huber
Max Features	log2(n_features)
Max Depth	5
Max Samples Split	100
Min Samples Leaf	39

Table 4: Optimal parameters for GB regression

Features	adaBoost
Loss	Linear
n_estimators	357

Table 5: Optimal parameters for adaBoost regression

Features	Extra Trees
n_estimators	400
Min Samples Split	2
Max Features	n_features
Max Depth	100
Bootstrap	True

Table 6: *Optimal parameters for ET*

Following the choice of the optimal parameters, the algorithm retrained and ran the test sets to evaluate the performance, as has been analyzed in section 4.3. The last part of the code produced the graph shown in figure 4, which is the key depiction of the overall project outcome in seeking to find the best algorithm to fit and model SOC levels in the given study area.

From a first look, three algorithms can be clearly distinguished from the rest. RF, GB and ET in green, dark gray and yellow respectively, have significantly lower errors and higher R^2 scores. All three algorithms almost match in performance with Random Forests proving to be the superior algorithm by only a small margin. In the bars below, it is seen how all three exactly match in MAPE at a value of 12%, while GB and ET match in determination coefficient with R^2 equal to 0.69 and RF only slightly above at 0.71.

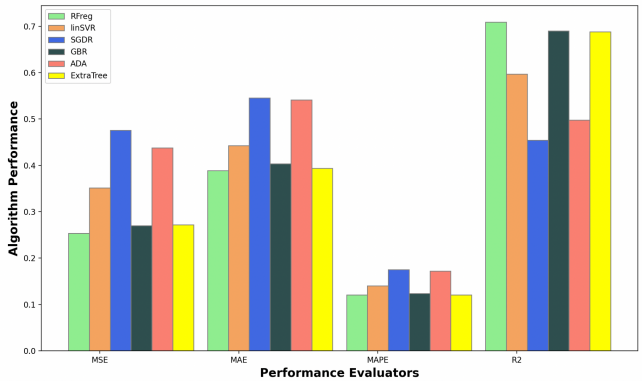

Figure 4: *Bar plot of algorithm performance (see legend) with MSE, MAE, MAPE and R^2 in order from left to right*

The MSE is also lowest for RF at 0.25 with both GBR and ET at 0.27, while MAE stood at 0.39 for RF and ET and 0.40 for GB. The overall comparison of the results between the top three algorithms is depicted in table 7 below.

Rank	MAE	MSE	MAPE	R2	Overall
1	RF, ET	RF	All	RF	RF
2	GBR	ET, GBR		ET, GBR	ET
3					GBR

Table 7: *Performance ranking comparison*

Across all evaluation methods, RF was at the top, hence has been ranked first. However, the last metric that was taken into account and which is not depicted above, was the training runtime. RF ran significantly faster at 0.3 seconds, with ET lagging at 0.6 seconds and GB at 0.7 seconds. Although these runtimes are evidently very small, the small dataset size has to be taken into account. Given these margins, with RF running at half the time of the other two algorithms, in possible future applications with a much larger dataset, or in the case of running in real-time, this runtime advantage of RF could be of great importance.

6 Conclusion

The scope of this project was to produce new knowledge and understanding in using ML methods to monitor soil characteristics, and in this case SOC levels. Given the limited availability of data that we had access to, we were successful in implementing various algorithms and optimizing them to accurately predict SOC. Given the initial performance of the algorithms without any feature extraction or parameter tuning, the R2 score was increased from the low 0.30's all the way up to 0.70's. This is an indication that one of our main objectives was a success, as our processing had a positive impact.

At no point was it expected that our optimization would be so great that it would achieve very high R2 scores close to 1, since the limited number of samples, at 840, were far too little for any algorithm to train that well. All six algorithms performed well in accuracy with MAPE ranging between 12-18%, which is generally considered moderately accurate, but very much acceptable.

The Random Forest regression has been determined to be the best performing algorithm across all the evaluation methods that were performed, and was also the most time-efficient. However, its lead margins over the second and third place algorithms were far too small to neglect Extra Trees and Gradient Boosting regressions. As will be discussed further in section 8 on future plans, all three algorithms will be essential in further research on SOC modeling, especially when testing different study areas with new climatic and topographic characteristics.

7 Future Plans

One of the research questions that we hoped to investigate but did not pursue due to time-constraints, was performing the same algorithm optimization and performance comparison across two study areas, at varying locations and climatic conditions. The goal would be to observe how location and environment play a role in the modeling of soil characteristics, and how ML models adapt to such differences. This is a very important question to answer, especially if remote sensing to model SOC is to be used on a global scale with confidence in its accuracy. The three top performing algorithms we found, RF, ET and

GB would be a very solid starting point in such an investigation, however, it is possible that other algorithms, or perhaps neural networks which were not investigated in this project, could prove to perform better.

8 References

Zhou, et al. “High-Resolution Digital Mapping of Soil Organic Carbon and Soil Total Nitrogen Using DEM Derivatives, Sentinel-1 and Sentinel-2 Data Based on Machine Learning Algorithms.” *Science of The Total Environment*, Elsevier, 13 Apr. 2020

Ma, Guolin, et al. “Digital Mapping of Soil Salinization Based on Sentinel-1 and Sentinel-2 Data Combined with Machine Learning Algorithms.” *Regional Sustainability*, Elsevier, 9 July 2021

Geurts, P., Ernst, D. & Wehenkel, L. Extremely randomized trees. Mach Learn 63, 3–42 (2006).

Gholizadeh, A., Žižala, D., Saberioon, M., Borůvka, L., 2018. Soil organic carbon and texture retrieving and mapping using proximal, airborne and Sentinel-2 spectral imaging. Re- mote Sens. Environ. 218, 89–103.

Goetz, Scott J, et al. “Mapping and Monitoring Carbon Stocks with Satellite Observations: A Comparison of Methods - Carbon Balance and Management.” BioMed Central, Springer International Publishing, 25 Mar. 2009.

Koehrsen, Will. “Hyperparameter Tuning the Random Forest in Python.” Medium, Towards Data Science, 10 Jan. 2018.

“3.3. Metrics and Scoring: Quantifying the Quality of Predictions.” Scikit.

9 Appendix

Please find our code and further reading in the gitlab through the link below:
https://gitlab.eecs.umich.edu/frilis/soc-mod